APPENDIX A

```
/*****************************************************************************
* $Header:   G:/PVCS/CFG/VCS/TA/SC/SRC/SHDEBCRE.H_V   1.2   15 Oct 1998 14:10:30
AMARAL  $
*
* Copyright (c) Cubic Transportation Systems, 1997-1998.  All Rights Reserved.
*
* File: shdebcre.h
*
* Desc: Debit/Credit Manager interface routines header file.
*
*****************************************************************************
* Version Control Information:
*
* $Log:   G:/PVCS/CFG/VCS/TA/SC/SRC/SHDEBCRE.H_V  $
*
*    Rev 1.2   15 Oct 1998 14:10:30   AMARAL
* Added prototype for DBCRDT_ComsStatus() and DBCRDT_COMMS_STATUS enum.
*
*    Rev 1.1   24 Jul 1998 18:11:26   AMARAL
* Added parameters to DBCRDT_Init() for DEV_SMADS.
*
*    Rev 1.0   05 Mar 1998 11:21:52   AMARAL
* Initial revision.
*****************************************************************************/


#ifndef SHDEBCRE_H
#define SHDEBCRE_H                          /* Include only once */


/* Include Files */


/* Defines */


/* Typedefs */
typedef enum CE_DBCRDT_ERRORS
{
    DBCRDT_NO_ERROR = 0,
    DBCRDT_AC_COMMS_UP,
    DBCRDT_AC_COMMS_DOWN,
    DBCRDT_DSM_GET_ERROR,
    DBCRDT_DSM_INIT_ERROR,
    DBCRDT_DSM_MSG_ERROR,
    DBCRDT_DSM_SAVE_ERROR,
    DBCRDT_DSM_UPDATE_ERROR,
    DBCRDT_DSM_VERSION_ERROR,
    DBCRDT_MEMORY_ERROR,
    DBCRDT_QUEUE_CLOSE_ERROR,
    DBCRDT_QUEUE_CREATE_ERROR,
    DBCRDT_QUEUE_EMPTY_MSG,
    DBCRDT_QUEUE_QUERY_ERROR,
    DBCRDT_QUEUE_READ_ERROR,
    DBCRDT_QUEUE_WRITE_ERROR,
    DBCRDT_SEM_CLOSE_ERROR,
    DBCRDT_SEM_CREATE_ERROR,
```

```
    DBCRDT_SEM_POST_ERROR,
    DBCRDT_SEM_RELEASE_ERROR,
    DBCRDT_SEM_REQUEST_ERROR,
    DBCRDT_SEM_RESET_ERROR,
    DBCRDT_SEM_TIMEOUT_ERROR,
    DBCRDT_THREAD_START_ERROR,
    DBCRDT_MAX_ERRORS
} DBCRDT_ERRORS;

typedef enum CE_DBCRDT_COMMS_STATUS
{
    DBCRDT_COMMS_UP = 0,
    DBCRDT_COMMS_R_LINE_UP,
    DBCRDT_COMMS_DOWN
} DBCRDT_COMMS_STATUS;


/* Global Variables */


/* Prototypes */
#if defined(__cplusplus)
extern "C"
{
#endif
DBCRDT_COMMS_STATUS DBCRDT_ComsStatus(VOID);
VOID    DBCRDT_ACComsDisable(VOID);
BOOL    DBCRDT_ACComsDisabled(VOID);
VOID    DBCRDT_ACComsEnable(VOID);
BOOL    DBCRDT_ClearQue(HQUEUE hQueue);
BOOL    DBCRDT_Init(USHORT usMajorVer, USHORT usMinorRev
#ifdef DEV_SMADS
                    , INT iHalfSecWaitsForMack
                    , INT iSecondsForResend
#endif
                   );
VOID    DBCRDT_PercentFree(PLONG plDsmPercentFree);
BOOL    DBCRDT_SendToHost(PUCHAR pszMsg, ULONG ulMsgLength, PUCHAR auchInfo);
VOID    DBCRDT_Stop(VOID);

/* Call back functions to be supplied by the calling application */
VOID    DBCRDT_Display(PCHAR szDispStr);
VOID    DBCRDT_DSM_Created(VOID);
VOID    DBCRDT_Error(INT iErrorID, ULONG ulErrorCode,
                     PCHAR szFileName, SHORT sLineNo);
BOOL    DBCRDT_SendToDevice(PUCHAR pszMsg, USHORT usMsgLength,
                            PUCHAR auchInfo);
#if defined(__cplusplus)
}
#endif

#endif
```

APPENDIX B

```
/*****************************************************************************
* $Header:   G:/pvcs/cfg/vcs/ta/smads/src/shdebcre.c_v   1.6   28 Jan 2000
18:14:00   DYOUNG  $
*
* Copyright (c) Cubic Transportation Systems, 1997-1999.  All Rights Reserved.
*
* File: shdebcre.c
*
* Desc: Debit/Credit Manager interface routines.
*
*****************************************************************************
* Version Control Information:
*
* $Log:   G:/pvcs/cfg/vcs/ta/smads/src/shdebcre.c_v  $
*
*    Rev 1.6   28 Jan 2000 18:14:00   DYOUNG
* Check for EUB messages after wakeup to send old msgs.
*
*    Rev 1.12   26 Jan 2000 16:10:42   AMARAL
* Added support for EUBx messages for TRANSIT AUTHORITY.
*
*    Rev 1.11   05 Jan 2000 18:52:04   amaral
* Put type in MACK for unknown message types.
*
*    Rev 1.10   25 Oct 1999 16:22:38   AMARAL
* Added check for sent messages when sending to central computer.
*
*    Rev 1.9   07 Jun 1999 17:14:58   AMARAL
* AR#158 Debit/Credit module shutdown causes protection a violation.
*
*    Rev 1.8   13 May 1999 14:13:42   AMARAL
* AR#138 Fixed EUC2 MACK verification.
*
*    Rev 1.7   07 May 1999 14:45:02   AMARAL
* Added processing to handle unexpected MACK EUC2 messages.
*
*    Rev 1.6   02 Mar 1999 11:26:24   amaral
* Only resend messages if less than 5 messages on queue to start with.
*
*    Rev 1.5   15 Oct 1998 16:12:18   AMARAL
* Added DBCRDT_ComsStatus() to report D/C AC comms status.
* Implemented controlled thread shutdown.
*
*    Rev 1.4   27 Jul 1998 15:44:08   AMARAL
* Corrected transmit length in SH_DCXmit() call in DBCRDT_ACTransmit().
*
*    Rev 1.3   24 Jul 1998 18:14:10   AMARAL
* Changed DBCRDT_ACTransmit() to not disable comms on failed message sent.
*
*    Rev 1.2   20 May 1998 09:04:04   amaral
* Change MACK status for EUC2 from MSG_ACKSTS_BUSY to MSG_ACKSTS_ABORT.
*
*    Rev 1.1   14 Apr 1998 11:24:38   AMARAL
* Added wait for MACKEUC2 from end device.
*
*    Rev 1.0   04 Mar 1998 16:09:08   AMARAL
* Initial revision.
```

```
******************************************************************************/


/* Include Files */
#define INCL_DOS
#define INCL_BASE
#define INCL_DOSERRORS
#define INCL_NOPM
#include <os2.h>

#include <stdlib.h>
#include <stdio.h>
#include <string.h>

#include "shdsmpub.h"
#include "shapc000.h"
#include "shmsg000.h"
#include "shmsg005.h"
#ifdef DEV_SMADS
#include "shmsg006.h"
#endif
#include "shlib000.h"
#include "sherr000.h"
#include "shtime00.h"
#include "shdebcre.h"


/* Defines */
#define DC_RECVQ_QUE_NAME       "\\QUEUES\\DBCRDT\\DBCRDTRX.QUE"
#define DC_XMITQ_QUE_NAME       "\\QUEUES\\DBCRDT\\DBCRDTTX.QUE"
#define DC_NO_SEMHANDLE         0UL
#define DC_DSM_FILENAME         "DEBTCRDT"
#ifdef DEV_SMADS
#define DC_DSM_FILESIZE         (2L * 1024L * 1024L)
#else
#define DC_DSM_FILESIZE         (16L * 1024L * 1024L)
#endif
#define DC_DSM_SEM_TIMEOUT      5000L
#define DC_MACK_ERROR           0xFF
#define DC_STACK_SIZE           40960
#define DC_INIT_TIMEOUT         3500L
#ifdef DEV_SMADS
#define WHITE_BLACK
#define BLUE_BROWN
#define RED_CYAN
#define BLACK_BACK
#define RED_BACK
#define BROWN_BACK
#define CYAN_BACK
#else
#define WHITE_BLACK             "\033[40;37m"
#define BLUE_BROWN              "\033[43;34m"
#define RED_CYAN                "\033[46;31m"
#define BLACK_BACK              "\033[40m"
#define RED_BACK                "\033[41m"
#define BROWN_BACK              "\033[43m"
#define CYAN_BACK               "\033[46m"
```

```
#endif


/* Typedefs */
typedef enum CE_DBCRDT_MSGS
{
    DC_MSG_NONE = 0,
    DC_MSG_SEND,
    DC_MSG_EXIT
} DBCRDT_MSGS;

typedef struct CS_DBCRDT_GENERIC
{
   MSG_C_HEADER header;
   CHAR         data[MSG_MAX_LENGTH - sizeof(MSG_C_HEADER)];
} DBCRDT_GENERIC, *PDBCRDT_GENERIC;


/* Global Variables */
static INT                  DC_iRecvMsgsId, DC_iXmitMsgsId, DC_iManagDSMId;
#ifdef DEV_SMADS
static INT                  DC_iHalfSecWaitsForMack, DC_iSecondsForResend;
#endif
static BOOL                 DC_bRunning;
static PVOID                DC_pCBData;
static HEV                  DC_InitSem, DC_ACXmitSem;
static HEV                  DC_RecvEndSem, DC_XmitEndSem, DC_DSMEndSem;
static HMTX                 DC_DSMSem;
static HQUEUE               DC_RecvQ, DC_XmitQ;
static PDBCRDT_GENERIC      pDC_sExtMsg, pDC_sRcvdMsg;
static DBCRDT_COMMS_STATUS  DC_eCommsStatus;


/* Prototypes */
static BOOL     DBCRDT_DSMAdd(PUCHAR pMsgPtr, SHORT sMsgLength,
                              PUCHAR auchInfo);
static UCHAR    DBCRDT_GetLIBResult(ULONG lrc);
static BOOL     DBCRDT_GetQueueMsg(HQUEUE hQueue, PUCHAR pszMsg,
                                   PULONG pulMsgLength, PULONG pulMsgID);
VOID ENV_CDECL  DBCRDT_ManageDSM(PVOID dummy);
static UCHAR    DBCRDT_ProcessEC03Msg(MSG_EC03 *psEC03Msg);
VOID ENV_CDECL  DBCRDT_RecvMsgs(PVOID dummy);
static USHORT   DBCRDT_Verify_MACK(PVOID pvExtMsg, USHORT usExtMsgLen,
                                   MSG_C_HEADER *ptMsgHdr, PUCHAR auchInfo,
                                   AC_ACCOM_MODE eMode);
VOID ENV_CDECL  DBCRDT_XmitMsgs(PVOID dummy);

/* Call back functions for DSM */
BOOL ENV_CDECL  DC_DSMRequestSem(VOID);
BOOL ENV_CDECL  DC_DSMReleaseSem(VOID);


/*****************************************************************************
* Function:         Debit/Credit Communications Status
* Desc:             Get the Debit/Credit Communications Status.
* Inputs:           N/A
* Outputs:          Return the Debit/Credit Communications Status.
```

```
* Return Value:     N/A
* External Effects: N/A
* Implementation:   N/A
*****************************************************************************/
DBCRDT_COMMS_STATUS DBCRDT_ComsStatus(VOID)
{
    return DC_eCommsStatus;
}


/*****************************************************************************
* Function:          Debit/Credit AC Comms Disabled
* Desc:              Test the DC_ACXmitSem.
* Inputs:            N/A
* Outputs:           Return TRUE if the semaphore is reset, else return FALSE.
* Return Value:      N/A
* External Effects:  N/A
* Implementation:    N/A
*****************************************************************************/
BOOL DBCRDT_ACComsDisabled(VOID)
{
    return (DosWaitEventSem(DC_ACXmitSem, 0L) == ERROR_TIMEOUT);
}


/*****************************************************************************
* Function:          Debit/Credit AC Comms Enable
* Desc:              Enable write to AC transmit queue.
* Inputs:            N/A
* Outputs:           N/A
* Return Value:      N/A
* External Effects:  N/A
* Implementation:    N/A
*****************************************************************************/
VOID DBCRDT_ACComsEnable(VOID)
{
    APIRET  rc;

    if (DBCRDT_ACComsDisabled())
    {
        DC_eCommsStatus = DBCRDT_COMMS_UP;
        DBCRDT_Display("DBCRDT_ACComsEnable: Area Controller On-Line.\r\n");
#ifndef DEV_SMADS
        DBCRDT_Error(DBCRDT_AC_COMMS_UP, 0UL, __FILE__, __LINE__);
#endif
        rc = DosPostEventSem(DC_ACXmitSem);
        if (rc && (rc != ERROR_ALREADY_POSTED))
        {
            DBCRDT_Error(DBCRDT_SEM_POST_ERROR, rc, __FILE__, __LINE__);
        }
    }
    return;
}


/*****************************************************************************
* Function:          Debit/Credit AC Comms Disable
```

```
* Desc:             Disable write to AC transmit queue.
* Inputs:           N/A
* Outputs:          N/A
* Return Value:     N/A
* External Effects: N/A
* Implementation:   N/A
*****************************************************************************/
VOID DBCRDT_ACComsDisable(VOID)
{
    APIRET  rc;
    ULONG   ulPostCount;

    if (DBCRDT_ACComsDisabled())        /* AC Comms are already disabled */
        return;

    DC_eCommsStatus = DBCRDT_COMMS_DOWN;
    DBCRDT_Display("DBCRDT_ACComsDisable: Area Controller Off-Line.\r\n");
#ifndef DEV_SMADS
    DBCRDT_Error(DBCRDT_AC_COMMS_DOWN, 0UL, __FILE__, __LINE__);
#endif
    rc = DosResetEventSem(DC_ACXmitSem, &ulPostCount);
    if (rc && (rc != ERROR_ALREADY_RESET))
    {
        DBCRDT_Error(DBCRDT_SEM_RESET_ERROR, rc, __FILE__, __LINE__);
    }
    return;
}


/*****************************************************************************
* Function:         Clear Debit/Credit Receive Queue
* Desc:             Clear Debit/Credit receive queue.
* Inputs:           N/A
* Outputs:          N/A
* Return Value:     N/A
* External Effects: N/A
* Implementation:   N/A
*****************************************************************************/
BOOL DBCRDT_ClearQue(HQUEUE hQueue)
{
    APIRET      rc;
    ULONG       ulMsgsInQ, ulMsgLength;
    BYTE        bMsgPriority;
    PVOID       pvMsgAddr;
    REQUESTDATA PidData;

    rc = DosQueryQueue(hQueue, &ulMsgsInQ);
    if (rc)
    {
        DBCRDT_Error(DBCRDT_QUEUE_QUERY_ERROR, rc, __FILE__, __LINE__);
        return FALSE;
    }

    while (ulMsgsInQ)   /* While there are messages in the queue */
    {   /* Read the queue and free the message memory */
        rc = DosReadQueue(hQueue, &PidData, &ulMsgLength, &pvMsgAddr, 0L,
                          DCWW_WAIT, &bMsgPriority, DC_NO_SEMHANDLE);
```

```
        if (rc)
        {
            DBCRDT_Error(DBCRDT_QUEUE_READ_ERROR, rc, __FILE__, __LINE__);
            return FALSE;
        }
        /* Free the message buffer */
        free(pvMsgAddr);
        ulMsgsInQ--;
    }
    return TRUE;
}   /* DBCRDT_ClearQue */


/*****************************************************************************
* Function:         Add Message To DSM
* Desc:             Add Debit/Credit message to DSM.
* Inputs:           N/A
* Outputs:          N/A
* Return Value:     N/A
* External Effects: N/A
* Implementation:   N/A
*****************************************************************************/
static BOOL DBCRDT_DSMAdd(PUCHAR pMsgPtr, SHORT sMsgLength, PUCHAR auchInfo)
{
    DSM_FILE_ERRORS rc;
    ULONG           ulKey, ulDatimStamp;
    LONG            lMsgFP;
    DSM_CBUF_HDR    tDiskMsgHdr;

    ulKey = ((MSG_C_HEADER *) pMsgPtr)->cics_trans_no.v;
    if (DSM_SearchKey(DC_pCBData, ulKey, &tDiskMsgHdr, &lMsgFP) != DSM_MSG_OK)
    {   /* Not found, save it */
        ulDatimStamp = SHTIM_time();
        rc = DSM_MsgSave(DC_pCBData, pMsgPtr, DSM_MSG_NONPRIORITY, auchInfo,
                         ulKey, sMsgLength, ulDatimStamp, &lMsgFP);
        if (rc != DSM_FILE_OK)
            DBCRDT_Error(DBCRDT_DSM_SAVE_ERROR, rc, __FILE__, __LINE__);
    }
    else                                /* Duplicate already exists */
    {
        rc = DSM_FILE_NO_ROOM_ERROR;    /* Set error so message doesn't xmit */
        DBCRDT_Display("DBCRDT_DSMAdd: Duplicate message ignored.\r\n");
    }
    return (rc == DSM_FILE_OK);
}   /* DBCRDT_DSMAdd */


/*****************************************************************************
* Function:         DSM Request Semaphore
* Desc:             Request semaphore for DSM access.
* Inputs:           N/A
* Outputs:          N/A
* Return Value:     N/A
* External Effects: N/A
* Implementation:   N/A
*****************************************************************************/
BOOL ENV_CDECL DC_DSMRequestSem(VOID)
```

```
{
    APIRET  rc;

    rc = DosRequestMutexSem(DC_DSMSem, DC_DSM_SEM_TIMEOUT);
    if (rc)
    {
        DBCRDT_Error(DBCRDT_SEM_REQUEST_ERROR, rc, __FILE__, __LINE__);
        return TRUE;
    }
    return FALSE;
}   /* DC_DSMRequestSem */


/*****************************************************************************
* Function:         DSM Release Semaphore
* Desc:             Release semaphore for DSM access.
* Inputs:           N/A
* Outputs:          N/A
* Return Value:     N/A
* External Effects: N/A
* Implementation:   N/A
*****************************************************************************/
BOOL ENV_CDECL DC_DSMReleaseSem(VOID)
{
    APIRET  rc;

    rc = DosReleaseMutexSem(DC_DSMSem);
    if (rc)
    {
        DBCRDT_Error(DBCRDT_SEM_RELEASE_ERROR, rc, __FILE__, __LINE__);
        return TRUE;
    }
    return FALSE;
}   /* DC_DSMReleaseSem */


/*****************************************************************************
* Function:         Debit/Credit Get Queue Message
* Desc:             Get a Debit/Credit message of the given queue.
* Inputs:           HQUEUE hQueue      - Handle of queue to get message from.
*                   PUCHAR pszMsg      - Pointer of buffer to place message.
*                   PULONG ulMsgLength - Pointer to store message data length.
* Outputs:          N/A
* Return Value:     TRUE if message retrieved else FALSE.
* External Effects: N/A
* Implementation:   N/A
*****************************************************************************/
static BOOL DBCRDT_GetQueueMsg(HQUEUE hQueue, PUCHAR pszMsg,
                               PULONG pulMsgLength, PULONG pulMsgID)
{
    APIRET      rc;
    ULONG       ulMsgsInQ;
    BOOL        bRc;
    UCHAR       ucMsgPriority;
    PVOID       pvMsgAddr;
    REQUESTDATA PidData;
```

```
    bRc = FALSE;
    rc = DosQueryQueue(hQueue, &ulMsgsInQ);
    if (rc)
        DBCRDT_Error(DBCRDT_QUEUE_QUERY_ERROR, rc, __FILE__, __LINE__);
    else if (ulMsgsInQ)
    {   /* Get message fromn Debit/Credit receive queue */
        rc = DosReadQueue(hQueue, &PidData, pulMsgLength, &pvMsgAddr, 0L,
                          DCWW_WAIT, &ucMsgPriority, DC_NO_SEMHANDLE);
        if (rc)
            DBCRDT_Error(DBCRDT_QUEUE_READ_ERROR, rc, __FILE__, __LINE__);
        else
        {   /* Free the response message buffer */
            *pulMsgID = PidData.ulData;
            memcpy(pszMsg, pvMsgAddr, *pulMsgLength);
            free(pvMsgAddr);
            bRc = TRUE;                 /* Message retrieved */
        }
    }
    return bRc;
}   /* DBCRDT_GetQueueMsg */


/*****************************************************************************
* Function:         Debit/Credit Initialize
* Desc:             Initialize Debit/Credit routines.
* Inputs:           N/A
* Outputs:          N/A
* Return Value:     N/A
* External Effects: N/A
* Implementation:   N/A
*****************************************************************************/
BOOL DBCRDT_Init(USHORT usMajorVer, USHORT usMinorRev
#ifdef DEV_SMADS
                , INT iHalfSecWaitsForMack
                , INT iSecondsForResend
#endif
)
{
    APIRET          rc;
    ULONG           ulPostCount;
    DSM_FILE_ERRORS dsmRc;

    DC_eCommsStatus = DBCRDT_COMMS_DOWN;

    /* Initialize memory */
    rc = DosAllocSharedMem((PVOID) &pDC_sExtMsg, NULL, sizeof(DBCRDT_GENERIC),
                           PAG_COMMIT | OBJ_GIVEABLE | PAG_WRITE);
    if (rc)
    {
        DBCRDT_Error(DBCRDT_MEMORY_ERROR, rc, __FILE__, __LINE__);
        return FALSE;
    }
    rc = DosAllocSharedMem((PVOID) &pDC_sRcvdMsg, NULL, sizeof(DBCRDT_GENERIC),
                           PAG_COMMIT | OBJ_GIVEABLE | PAG_WRITE);
    if (rc)
    {
        DBCRDT_Error(DBCRDT_MEMORY_ERROR, rc, __FILE__, __LINE__);
```

```
    return FALSE;
}

/* Initialize semaphores */
rc = DosCreateEventSem(NULL, &DC_InitSem, 0L, FALSE);
if (rc)
{
    DBCRDT_Error(DBCRDT_SEM_CREATE_ERROR, rc, __FILE__, __LINE__);
    return FALSE;
}
rc = DosCreateEventSem(NULL, &DC_ACXmitSem, 0L, TRUE);
if (rc)
{
    DBCRDT_Error(DBCRDT_SEM_CREATE_ERROR, rc, __FILE__, __LINE__);
    return FALSE;
}
rc = DosCreateMutexSem(NULL, &DC_DSMSem, 0L, FALSE);
if (rc)
{
    DBCRDT_Error(DBCRDT_SEM_CREATE_ERROR, rc, __FILE__, __LINE__);
    return FALSE;
}
rc = DosCreateEventSem(NULL, &DC_RecvEndSem, 0L, FALSE);
if (rc)
{
    DBCRDT_Error(DBCRDT_SEM_CREATE_ERROR, rc, __FILE__, __LINE__);
    return FALSE;
}
rc = DosCreateEventSem(NULL, &DC_XmitEndSem, 0L, FALSE);
if (rc)
{
    DBCRDT_Error(DBCRDT_SEM_CREATE_ERROR, rc, __FILE__, __LINE__);
    return FALSE;
}
rc = DosCreateEventSem(NULL, &DC_DSMEndSem, 0L, FALSE);
if (rc)
{
    DBCRDT_Error(DBCRDT_SEM_CREATE_ERROR, rc, __FILE__, __LINE__);
    return FALSE;
}

/* Initialize queues */
rc = DosCreateQueue(&DC_RecvQ, QUE_FIFO, DC_RECVQ_QUE_NAME);
if (rc)
{
    DBCRDT_Error(DBCRDT_QUEUE_CREATE_ERROR, rc, __FILE__, __LINE__);
    return FALSE;
}
rc = DosCreateQueue(&DC_XmitQ, QUE_PRIORITY, DC_XMITQ_QUE_NAME);
if (rc)
{
    DBCRDT_Error(DBCRDT_QUEUE_CREATE_ERROR, rc, __FILE__, __LINE__);
    return FALSE;
}

/* Initialize DSM */
do
```

```
{
    dsmRc = DSM_CBufInit(&DC_pCBData, DC_DSM_FILENAME, DC_DSM_FILESIZE,
                         FALSE, usMajorVer, usMinorRev,
                         DC_DSMRequestSem, DC_DSMReleaseSem);
    if (dsmRc == DSM_FILE_WRONG_VERSION)
    {
        DBCRDT_Error(DBCRDT_DSM_VERSION_ERROR, dsmRc, __FILE__, __LINE__);
        DBCRDT_Display("DBCRDT_Init: DSM is wrong version, "
                       "updating version.\r\n");
        DSM_UpdateVersion(DC_pCBData, usMajorVer, usMinorRev, FALSE);
    }
} while (dsmRc == DSM_FILE_WRONG_VERSION);
if (dsmRc == DSM_FILE_CREATED)
    DBCRDT_DSM_Created();
else if (dsmRc != DSM_FILE_OPENED)
{
    DBCRDT_Error(DBCRDT_DSM_INIT_ERROR, dsmRc, __FILE__, __LINE__);
    return FALSE;
}

/* Start threads */
DC_bRunning = TRUE;
rc = DosResetEventSem(DC_InitSem, &ulPostCount);
if (rc && rc != ERROR_ALREADY_RESET)
{
    DBCRDT_Error(DBCRDT_SEM_RESET_ERROR, rc, __FILE__, __LINE__);
    return FALSE;
}
DC_iRecvMsgsId = _beginthread(DBCRDT_RecvMsgs, NULL, DC_STACK_SIZE, NULL);
if (DC_iRecvMsgsId == -1)
{
    DBCRDT_Error(DBCRDT_THREAD_START_ERROR, (ULONG) DC_iRecvMsgsId,
                 __FILE__, __LINE__);
    return FALSE;
}
rc = DosWaitEventSem(DC_InitSem, DC_INIT_TIMEOUT);
if (rc)
{
    DBCRDT_Error(DBCRDT_SEM_TIMEOUT_ERROR, rc, __FILE__, __LINE__);
    return FALSE;
}

rc = DosResetEventSem(DC_InitSem, &ulPostCount);
if (rc && rc != ERROR_ALREADY_RESET)
{
    DBCRDT_Error(DBCRDT_SEM_RESET_ERROR, rc, __FILE__, __LINE__);
    return FALSE;
}
DC_iXmitMsgsId = _beginthread(DBCRDT_XmitMsgs, NULL, DC_STACK_SIZE, NULL);
if (DC_iXmitMsgsId == -1)
{
    DBCRDT_Error(DBCRDT_THREAD_START_ERROR, (ULONG) DC_iXmitMsgsId,
                 __FILE__, __LINE__);
    return FALSE;
}
rc = DosWaitEventSem(DC_InitSem, DC_INIT_TIMEOUT);
if (rc)
```

```
    {
        DBCRDT_Error(DBCRDT_SEM_TIMEOUT_ERROR, rc, __FILE__, __LINE__);
        return FALSE;
    }

    rc = DosResetEventSem(DC_InitSem, &ulPostCount);
    if (rc && rc != ERROR_ALREADY_RESET)
    {
        DBCRDT_Error(DBCRDT_SEM_RESET_ERROR, rc, __FILE__, __LINE__);
        return FALSE;
    }
    DC_iManagDSMId = _beginthread(DBCRDT_ManageDSM, NULL, DC_STACK_SIZE, NULL);
    if (DC_iManagDSMId == -1)
    {
        DBCRDT_Error(DBCRDT_THREAD_START_ERROR, (ULONG) DC_iManagDSMId,
                     __FILE__, __LINE__);
        return FALSE;
    }
    rc = DosWaitEventSem(DC_InitSem, DC_INIT_TIMEOUT);
    if (rc)
    {
        DBCRDT_Error(DBCRDT_SEM_TIMEOUT_ERROR, rc, __FILE__, __LINE__);
        return FALSE;
    }

#ifdef DEV_SMADS
    /* Set up wait times passed in from application */
    DC_iHalfSecWaitsForMack = iHalfSecWaitsForMack;
    DC_iSecondsForResend = iSecondsForResend;
#endif

    DC_eCommsStatus = DBCRDT_COMMS_UP;
    return DC_bRunning;
}   /* DBCRDT_Init */


/*****************************************************************************
* Function:         Debit/Credit Percent Free
* Desc:             Determine percentage of free space available in DSM file.
* Inputs:           pulDsmFree is a pointer to store the percentage available
*                   of the Debit/Credit DSM files.
* Outputs:          The percentage available is an long value represented in
*                   hundredths of a percent, i.e. a value of 1 is 00.01% and a
*                   value of 10000 is 100.00%.  The valid range is -1 to
*                   10000, where -1 means the value could not be determined.
* Return Value:     N/A
* External Effects: N/A
* Implementation:   N/A
*****************************************************************************/
VOID DBCRDT_PercentFree(PLONG plDsmPercentFree)
{
    LONG    lFreeSpace;

    DSM_CBufUsage(DC_pCBData, &lFreeSpace);
    if (lFreeSpace == -1L)
        *plDsmPercentFree = -1L;
    else
```

```
        *plDsmPercentFree = (LONG) (((double) lFreeSpace * 10000) /
                                                        DC_DSM_FILESIZE);
    return;
}


/*****************************************************************************
* Function:         Debit/Credit Receive Messages
* Desc:             Receive Debit/Credit messages from host computer.  This
*                   thread handles EC03, EUC0, EUC2, and MACKEUC2 messages
*                   only.
* Inputs:           N/A
* Outputs:          N/A
* Return Value:     N/A
* External Effects: N/A
* Implementation:   N/A
*****************************************************************************/
VOID ENV_CDECL DBCRDT_RecvMsgs(PVOID dummy)
{
    APIRET          rc;
    INT             iLcv;
    USHORT          usErr, usLength, usIntLen;
    ULONG           ulRc, ulMsgLength, ulMsgsInQ;
    BOOL            bRunning, bDone;
    UCHAR           MsgPriority;
    UCHAR           szMsgType[MSG_ID_TYPE_MAX], szBuffer[82];
    UCHAR           szMsg[MSG_MAX_LENGTH], auchInfo[DSM_INFO_SIZE];
    PVOID           MsgAddr;
    REQUESTDATA     PidData;
    DBCRDT_GENERIC  sIntMsg;
    PDBCRDT_GENERIC pMsg = NULL;
    MSG_EUC2        *psEUC2Msg;
#ifdef DEV_SMADS
    MSG_EUB2        *psEUB2Msg;
#endif
    MSG_C_MACK      sMackMsg;
    AC_ACCOM_MODE   eMode;
    INT             iWait;              /* Number of half seconds to wait */
                                        /* for response in DC_RecvQ       */

    /* Local Initialization */
    bRunning = TRUE;

    rc = DosAllocSharedMem((PVOID) &pMsg, NULL, sizeof(DBCRDT_GENERIC),
                           PAG_COMMIT | OBJ_GIVEABLE | PAG_WRITE);
    if (rc)
    {
        DBCRDT_Error(DBCRDT_MEMORY_ERROR, 0UL, __FILE__, __LINE__);
        bRunning = FALSE;
    }

    /* Post 'Initialized' semaphore */
    rc = DosPostEventSem(DC_InitSem);
    if (rc && (rc != ERROR_ALREADY_POSTED))
    {
        DBCRDT_Error(DBCRDT_SEM_POST_ERROR, rc, __FILE__, __LINE__);
        if (bRunning)
```

```
    {
        DosFreeMem(pMsg);
        bRunning = FALSE;
    }
}

/* Init the AC recv mode */
eMode = AC_RECV_MODE;
while (bRunning && DC_bRunning)
{
    usErr = SH_DCRecv(pMsg, eMode, &usLength);
    if (usErr == AC_ERROR_OK)
    {
        if (usLength == 0)          /* Session was deallocated */
        {
            eMode = AC_RECV_MODE;
            usLength = sizeof(DBCRDT_GENERIC);

            /* Remove all messages from receive queue */
            DBCRDT_ClearQue(DC_RecvQ);
            continue;
        }

        /* Only EC03, EUC0, & EUC2 messages are received */
        /* Default to one second wait for response */
        iWait = 2;

        /* Receive any message from the AC then enable */
        /* transmit to AC if previously disabled.      */
        if (DBCRDT_ACComsDisabled())
        {
            DBCRDT_ACComsEnable();
        }
        DC_eCommsStatus = DBCRDT_COMMS_UP;
        usIntLen = usLength;
        ulRc = LIB_msg_byte_to_strc((PUCHAR) pMsg, &sIntMsg, &usIntLen);

        sprintf(szBuffer, RED_CYAN "Rx %4.4s Len 0x%x" WHITE_BLACK "\r\n",
                sIntMsg.header.type, usLength);
        DBCRDT_Display(szBuffer);

        /* Clear D/C receive queue of unexpected messages */
        rc = DosQueryQueue(DC_RecvQ, &ulMsgsInQ);
        if (rc)
        {
            DBCRDT_Error(DBCRDT_QUEUE_QUERY_ERROR, rc, __FILE__, __LINE__);
            ulMsgsInQ = 0UL;
        }
        while (ulMsgsInQ)
        {
            rc = DosReadQueue(DC_RecvQ, &PidData, &ulMsgLength,
                              &MsgAddr, 0L, DCWW_WAIT,
                              &MsgPriority, DC_NO_SEMHANDLE);
            if (rc)
            {
                DBCRDT_Error(DBCRDT_QUEUE_READ_ERROR, rc,
                             __FILE__, __LINE__);
```

```
            }
            else if (PidData.ulData == DC_MSG_EXIT)
            {
                bRunning = FALSE;
            }
            ulMsgsInQ--;
            free(MsgAddr);
        }

        /* Create MACK message */
        memcpy(sMackMsg.mack_id, MSG_ID_MACK, MSG_ID_TYPE_MAX);
        memcpy(sMackMsg.type, sIntMsg.header.type, MSG_ID_TYPE_MAX);
        sMackMsg.cics_trans_no.v = sIntMsg.header.cics_trans_no.v;
        sMackMsg.status = DBCRDT_GetLIBResult(ulRc);
        if (ulRc != LIB_SUCCESS)
        {
            LIB_cnv_ebcdic_to_ascii((PUCHAR) pMsg, (PUCHAR) &sIntMsg,
                                    MSG_ID_TYPE_MAX);
            DBCRDT_Error(DBCRDT_DSM_MSG_ERROR, ulRc, __FILE__, __LINE__);
            /* Transmit the non-zero MACK */
            LIB_msg_tx_mack(&sMackMsg, szMsg, &usIntLen);
            DBCRDT_SendToHost(szMsg, (ULONG) usIntLen, NULL);
        }
        else if (!strncmp(sIntMsg.header.type, MSG_ID_EC03,
                          MSG_ID_TYPE_MAX))
        {   /* Transmit the MACK EC03 */
            sMackMsg.status = DBCRDT_ProcessEC03Msg((MSG_EC03 *) &sIntMsg);
            LIB_msg_tx_mack(&sMackMsg, szMsg, &usIntLen);
            DBCRDT_SendToHost(szMsg, (ULONG) usIntLen, NULL);
        }
        else if (!strncmp(sIntMsg.header.type, MSG_ID_EUC0,
                          MSG_ID_TYPE_MAX))
        {   /* Transmit the MACK EUC0 */
            LIB_msg_tx_mack(&sMackMsg, szMsg, &usIntLen);
            DBCRDT_SendToHost(szMsg, (ULONG) usIntLen, NULL);
#ifdef DEV_SMADS
            DBCRDT_SendToDevice((PUCHAR) pMsg, MSG_ID_TYPE_MAX, auchInfo);
#endif

        }
        else if (!strncmp(sIntMsg.header.type, MSG_ID_EUC2,
                          MSG_ID_TYPE_MAX))
        {   /* Send EUC2 to end device */
            psEUC2Msg = (MSG_EUC2 *) &sIntMsg;
            /* This message originates at the AC.  The MACKEUC2 */
            /* message that is sent in response is put on the   */
            /* receive queue, DC_RecvQ.                         */
            LIB_cnv_ebcdic_to_ascii(
                      psEUC2Msg->fdata.common_iso_data_2.card_accp_term_id,
                      psEUC2Msg->fdata.common_iso_data_2.card_accp_term_id,
                      CARD_ACCP_TERMINAL_ID_MAX);
            /* Get the SCP from the terminal ID */
            auchInfo[0] =
             psEUC2Msg->fdata.common_iso_data_2.card_accp_term_id[4] - '0';
            auchInfo[1] =
             psEUC2Msg->fdata.common_iso_data_2.card_accp_term_id[5] - '0';
            auchInfo[2] =
```

```
                  ((psEUC2Msg->fdata.common_iso_data_2.card_accp_term_id[6] -
'0') * 10) +
                   psEUC2Msg->fdata.common_iso_data_2.card_accp_term_id[7] - '0';

                  /* Send EUC2 to device that sent the EUC1 or EUC6 */
                  if (!DBCRDT_SendToDevice((PUCHAR) pMsg, usLength, auchInfo))
                  {   /* Can't send to device, transmit a non-zero MACK */
                      sMackMsg.status = MSG_ACKSTS_ABORT;
                      LIB_msg_tx_mack(&sMackMsg, szMsg, &usIntLen);
                      DBCRDT_SendToHost(szMsg, (ULONG) usIntLen, NULL);
                  }
                  else
                  {   /* For EUC2, we should get a MACK back from the  */
                      /* device.  Make sure we wait long enough for    */
                      /* longest possible time allowed to get a        */
                      /* repsonse back from device, including possible */
                      /* re-transmit cases.                            */
#ifdef DEV_SMADS
                          iWait = DC_iHalfSecWaitsForMack;
#else
                          iWait = 60;
#endif
                  }
              }
#ifdef DEV_SMADS
              else if (!strncmp(sIntMsg.header.type, MSG_ID_EUB2,
                                MSG_ID_TYPE_MAX))
              {   /* Send EUB2 to end device */
                  psEUB2Msg = (MSG_EUB2 *) &sIntMsg;
                  /* This message originates at the AC.  The MACKEUB2 */
                  /* message that is sent in response is put on the   */
                  /* receive queue, DC_RecvQ.                         */
                  LIB_cnv_ebcdic_to_ascii(
                            &psEUB2Msg->fdata.tEUB_Hdr.card_accp_term_id[0],
                            &psEUB2Msg->fdata.tEUB_Hdr.card_accp_term_id[0],
                            MSG_EUC5_CARD_ACCP_ID_MAX);
                  /* Get the SCP from the terminal ID */
                  auchInfo[0] =
                   psEUB2Msg->fdata.tEUB_Hdr.card_accp_term_id[4] - '0';
                  auchInfo[1] =
                   psEUB2Msg->fdata.tEUB_Hdr.card_accp_term_id[5] - '0';
                  auchInfo[2] =
                   ((psEUB2Msg->fdata.tEUB_Hdr.card_accp_term_id[6] - '0') * 10) +
                     psEUB2Msg->fdata.tEUB_Hdr.card_accp_term_id[7] - '0';

                  /* Send EUB2 to device that sent the EUB1  */
                  if (!DBCRDT_SendToDevice((PUCHAR) pMsg, usLength, auchInfo))
                  {   /* Can't send to device, transmit a non-zero MACK */
                      sMackMsg.status = MSG_ACKSTS_ABORT;
                      LIB_msg_tx_mack(&sMackMsg, szMsg, &usIntLen);
                      DBCRDT_SendToHost(szMsg, (ULONG) usIntLen, NULL);
                  }
                  else
                  {   /* For EUB2, we should get a MACK back from the  */
                      /* device.  Make sure we wait long enough for    */
                      /* longest possible time allowed to get a        */
                      /* repsonse back from device, including possible */
```

```
                /* re-transmit cases.                                   */
                    iWait = DC_iHalfSecWaitsForMack;
            }
        }
#endif
        else    /* The message is unknown and not processed */
        {
            DBCRDT_Error(DBCRDT_DSM_MSG_ERROR, LIB_W_MSG_UNKNOWN,
                         __FILE__, __LINE__);
            /* Transmit the non-zero MACK */
            sMackMsg.status = MSG_ACKSTS_TRANSIDINV;
            LIB_msg_tx_mack(&sMackMsg, szMsg, &usIntLen);
            DBCRDT_SendToHost(szMsg, (ULONG) usIntLen, NULL);
        }

        eMode = AC_RECV_MODE;
        ulMsgsInQ = 0UL;
        bDone = FALSE;
        for (iLcv = 0; (iLcv < iWait) && !bDone; iLcv++)
        {   /* Check for a queue message for up to given halve seconds */
            rc = DosQueryQueue(DC_RecvQ, &ulMsgsInQ);
            if (rc)
            {
                DBCRDT_Error(DBCRDT_QUEUE_QUERY_ERROR, rc,
                             __FILE__, __LINE__);
                ulMsgsInQ = 0UL;
            }
            if (!ulMsgsInQ)
            {
                DosSleep(500L);
            }
            else
            {
                do
                {
                    rc = DosReadQueue(DC_RecvQ, &PidData, &ulMsgLength,
                                      &MsgAddr, 0L, DCWW_WAIT,
                                      &MsgPriority, DC_NO_SEMHANDLE);
                    if (rc)
                    {
                        DBCRDT_Error(DBCRDT_QUEUE_READ_ERROR, rc,
                                     __FILE__, __LINE__);
                    }
                    else if (PidData.ulData == DC_MSG_EXIT)
                    {
                        bRunning = FALSE;
                        bDone = TRUE;
                    }
                    else
                    {   /* If waiting for an EUB2/EUC2, verify MACK */
                        /* matches current EUB2/EUC2 message.  Also */
                        /* verify that the cics_trans_no.v matches  */
                        usLength = (USHORT) ulMsgLength;
                        memcpy(pMsg, MsgAddr, ulMsgLength);

                        /* Display Message header */
                        LIB_cnv_ebcdic_to_ascii(MsgAddr, szMsgType,
```

```
                                        MSG_ID_TYPE_MAX);
                    if (!strncmp(szMsgType, MSG_ID_MACK,
                                MSG_ID_TYPE_MAX))
                    {   /* Received a MACK */
                        if (LIB_msg_rx_mack((PUCHAR) MsgAddr,
                                        &sMackMsg,
                                        &usLength) == LIB_SUCCESS)
                        {   /* Verify the CICS # of the MACK */
                            if (sMackMsg.cics_trans_no.v ==
                                        sIntMsg.header.cics_trans_no.v)
                            {   /* This is the correct MACK */
                                sprintf(szBuffer, CYAN_BACK
                                        "Rx MACK %4.4s Status %d"
                                        BLACK_BACK "\r\n",
                                        sMackMsg.type,
                                        sMackMsg.status);
                                /* CICS numbers match */
                                bDone = TRUE;
                                /* Go back to top and send */
                                /* the response in pMsg    */
                                eMode = AC_XMIT_MODE;
                            }
                            else
                            {   /* Wrong MACK for current message */
                                sprintf(szBuffer, RED_BACK
                                        "Rx WRONG MACK %4.4s Status %d"
                                        BLACK_BACK "\r\n",
                                        sMackMsg.type,
                                        sMackMsg.status);
                            }
                        }
                        else
                        {   /* Invalid MACK received */
                            sprintf(szBuffer, CYAN_BACK "Rx MACK "
                                    "Unknown" BLACK_BACK "\r\n");
                        }
                    }
                    else
                    {   /* Display non-MACK message received */
                        sprintf(szBuffer, CYAN_BACK "Rx %4.4s"
                                BLACK_BACK "\r\n", szMsgType);
                    }
                    DBCRDT_Display(szBuffer);
                }
                ulMsgsInQ--;
                if (!rc)
                {   /* Free the response message buffer */
                    free(MsgAddr);
                }
            } while (ulMsgsInQ && !bDone);
        }
    }
    if (!bDone)
    {   /* Timed out */
        sprintf(szBuffer, RED_BACK
                "Timed out waiting for response to %4.4s." BLACK_BACK
                "\r\n", sIntMsg.header.type);
```

```
                DBCRDT_Display(szBuffer);
            }
        }
        else
        {
#if FALSE   /* Don't allow comms to get disabled */
            if (!DBCRDT_ACComsDisabled())  /* Comms enabled, disable it */
            {
                DBCRDT_ACComsDisable();
            }
#endif
            sprintf(szBuffer, RED_BACK "DBCRDT_RecvMsgs: SH_DCRecv error, "
                    "mode=%d, err=%d." BLACK_BACK "\r\n", eMode, usErr);
            DBCRDT_Display(szBuffer);
            DosSleep(30000L);           /* Sleep 30 seconds on receive error */
            /* Go back to a known state waiting to receive */
            eMode = AC_RECV_MODE;
        }
    }
    DosFreeMem(pMsg);

    DC_iRecvMsgsId = -1;

    /* Post 'Closing' semaphore */
    rc = DosPostEventSem(DC_RecvEndSem);
    if (rc && (rc != ERROR_ALREADY_POSTED))
    {
        DBCRDT_Error(DBCRDT_SEM_POST_ERROR, rc, __FILE__, __LINE__);
    }

    dummy = dummy;
    _endthread();
}   /* DBCRDT_RecvMsgs */


/*****************************************************************************
* Function:         Debit/Credit Stop
* Desc:             Stop Debit/Credit threads, release memory, and close queus.
* Inputs:           N/A
* Outputs:          N/A
* Return Value:     N/A
* External Effects: N/A
* Implementation:   N/A
*****************************************************************************/
VOID DBCRDT_Stop(VOID)
{
    APIRET  rc;
    ULONG   ulPriority, ulPostCount;

    DBCRDT_Display("DBCRDT_Stop: Stopping Debit/Credit Comms.\r\n");

    DBCRDT_ACComsDisable();             /* Stop AC comms */

    /* Stop threads */
    rc = DosResetEventSem(DC_RecvEndSem, &ulPostCount);
    if (rc && rc != ERROR_ALREADY_RESET)
        DBCRDT_Error(DBCRDT_SEM_RESET_ERROR, rc, __FILE__, __LINE__);
```

```
    rc = DosResetEventSem(DC_XmitEndSem, &ulPostCount);
    if (rc && rc != ERROR_ALREADY_RESET)
        DBCRDT_Error(DBCRDT_SEM_RESET_ERROR, rc, __FILE__, __LINE__);
    rc = DosResetEventSem(DC_DSMEndSem, &ulPostCount);
    if (rc && rc != ERROR_ALREADY_RESET)
        DBCRDT_Error(DBCRDT_SEM_RESET_ERROR, rc, __FILE__, __LINE__);

    /* Send signals to stop threads */
    DC_bRunning = FALSE;
    ulPriority = 15;
    rc = DosWriteQueue(DC_RecvQ, DC_MSG_EXIT, 0UL, NULL, ulPriority);
    if (rc)
        DBCRDT_Error(DBCRDT_QUEUE_WRITE_ERROR, rc, __FILE__, __LINE__);
    rc = DosWriteQueue(DC_XmitQ, DC_MSG_EXIT, 0UL, NULL, ulPriority);
    if (rc)
        DBCRDT_Error(DBCRDT_QUEUE_WRITE_ERROR, rc, __FILE__, __LINE__);

    /* Wait for threads to stop */
    rc = DosWaitEventSem(DC_DSMEndSem, DC_INIT_TIMEOUT);
    if (rc)
    {
        DBCRDT_Error(DBCRDT_SEM_TIMEOUT_ERROR, rc, __FILE__, __LINE__);
        if (DC_iManagDSMId != -1)
        {
            DosKillThread(DC_iManagDSMId);
            DBCRDT_Display("DBCRDT_Stop: Killed DSM thread!\r\n");
        }
    }
    rc = DosWaitEventSem(DC_XmitEndSem, DC_INIT_TIMEOUT);
    if (rc)
    {
        DBCRDT_Error(DBCRDT_SEM_TIMEOUT_ERROR, rc, __FILE__, __LINE__);
        if (DC_iXmitMsgsId != -1)
        {
            DosKillThread(DC_iXmitMsgsId);
            DBCRDT_Display("DBCRDT_Stop: Killed Transmit thread!\r\n");
        }
    }
    rc = DosWaitEventSem(DC_RecvEndSem, DC_INIT_TIMEOUT);
    if (rc)
    {
#ifndef DEV_SMADS
        DBCRDT_Error(DBCRDT_SEM_TIMEOUT_ERROR, rc, __FILE__, __LINE__);
#endif
        if (DC_iRecvMsgsId != -1)
        {
            DosKillThread(DC_iRecvMsgsId);
            DBCRDT_Display("DBCRDT_Stop: Killed Receive thread!\r\n");
        }
    }

    /* Close queues */
    DBCRDT_ClearQue(DC_RecvQ);          /* Release message memory on queue */
    rc = DosCloseQueue(DC_RecvQ);
    if (rc)
        DBCRDT_Error(DBCRDT_QUEUE_CLOSE_ERROR, rc, __FILE__, __LINE__);
    DBCRDT_ClearQue(DC_XmitQ);          /* Release message memory on queue */
```

```
    rc = DosCloseQueue(DC_XmitQ);
    if (rc)
        DBCRDT_Error(DBCRDT_QUEUE_CLOSE_ERROR, rc, __FILE__, __LINE__);

    /* Close semaphores */
    rc = DosCloseEventSem(DC_InitSem);
    if (rc)
        DBCRDT_Error(DBCRDT_SEM_CLOSE_ERROR, rc, __FILE__, __LINE__);
    rc = DosCloseEventSem(DC_ACXmitSem);
    if (rc)
        DBCRDT_Error(DBCRDT_SEM_CLOSE_ERROR, rc, __FILE__, __LINE__);
    rc = DosCloseMutexSem(DC_DSMSem);
    if (rc)
        DBCRDT_Error(DBCRDT_SEM_CLOSE_ERROR, rc, __FILE__, __LINE__);
    rc = DosCloseEventSem(DC_RecvEndSem);
    if (rc)
        DBCRDT_Error(DBCRDT_SEM_CLOSE_ERROR, rc, __FILE__, __LINE__);
    rc = DosCloseEventSem(DC_XmitEndSem);
    if (rc)
        DBCRDT_Error(DBCRDT_SEM_CLOSE_ERROR, rc, __FILE__, __LINE__);
    rc = DosCloseEventSem(DC_DSMEndSem);
    if (rc)
        DBCRDT_Error(DBCRDT_SEM_CLOSE_ERROR, rc, __FILE__, __LINE__);

    /* Release memory */
    DSM_Close(DC_pCBData);
    DosFreeMem(pDC_sExtMsg);
    DosFreeMem(pDC_sRcvdMsg);

    DBCRDT_Display("DBCRDT_Stop: Debit/Credit Comms Stopped!\r\n");
    return;
}   /* DBCRDT_Stop */


/*****************************************************************************
* Function:          Debit/Credit Send Message to Host
* Desc:              Send a message to the host Debit/Credit processor.
* Inputs:            PUCHAR pszMsg      - Pointer to messge to transmit.
*                        MACKEUC0 or MACKEC03 from Receive thread,
*                        MACKEUC2, EUC1, EUC3, EUC4, EUC5, or EUC6 from device
*                        (EV/AVM),
*                        EUB1, EUB3, EUB5,MACKEUC3, MACKEUC4, or MACKEUC5 from
*                        EV (TRANSIT AUTHORITY only).
*                    USHORT usMsgLength - Length of message data.
* Outputs:           N/A
* Return Value:      N/A
* External Effects: N/A
* Implementation:    N/A
*****************************************************************************/
BOOL DBCRDT_SendToHost(PUCHAR pszMsg, ULONG ulMsgLength, PUCHAR auchInfo)
{
    APIRET          rc;
    ULONG           ulPriority;
    BOOL            bRc;
    UCHAR           szMsgType[MSG_ID_TYPE_MAX];
    PUCHAR          pMsgPtr;
#ifdef DEV_SMADS
```

```
    LONG            lMsgFP;
    ULONG           ulKey;
    UCHAR           szBuffer[82];
    DSM_CBUF_HDR    tDiskMsgHdr;
    DSM_MSG_STATE   emRc;
#endif

    /* Place message on Debit/Credit transmit queue */
    pMsgPtr = (PUCHAR) malloc(ulMsgLength + DSM_INFO_SIZE);
    if (pMsgPtr == NULL)
    {
        DBCRDT_Error(DBCRDT_MEMORY_ERROR, 0UL, __FILE__, __LINE__);
        return FALSE;
    }

    LIB_cnv_ebcdic_to_ascii(pszMsg, szMsgType, MSG_ID_TYPE_MAX);
    if (!strncmp(szMsgType, MSG_ID_EUC3, MSG_ID_TYPE_MAX) ||
        !strncmp(szMsgType, MSG_ID_EUC4, MSG_ID_TYPE_MAX) ||
        !strncmp(szMsgType, MSG_ID_EUC5, MSG_ID_TYPE_MAX) ||
        !strncmp(szMsgType, MSG_ID_EUB3, MSG_ID_TYPE_MAX) ||
        !strncmp(szMsgType, MSG_ID_EUB5, MSG_ID_TYPE_MAX))
    {
        /* Place message in circular buffer file */
        if (!DBCRDT_DSMAdd(pszMsg, ulMsgLength, auchInfo))
        {
            free(pMsgPtr);
            return TRUE;                /* Duplicate or buffer full */
        }
        ulPriority = 0;
    }
    else    /* EUC0, EUC1, EUC6, MACKEC03, MACKEUC0, or MACKEUC2 */
    {
        ulPriority = 15;
    }
    /* Put message on host send queue */
    if (!strncmp(szMsgType, MSG_ID_MACK, MSG_ID_TYPE_MAX))  /* MACK */
    {
#ifdef DEV_SMADS
        /* For TRANSIT AUTHORITY, MACKS back to device are asynchronous, wait
for       */
        /* confirmation of delivery before completion of message in the DSM */
        LIB_cnv_ebcdic_to_ascii(pszMsg + MSG_ID_TYPE_MAX, szMsgType,
                                MSG_ID_TYPE_MAX);
        if (!strncmp(szMsgType, MSG_ID_EUC3, MSG_ID_TYPE_MAX) ||
            !strncmp(szMsgType, MSG_ID_EUC4, MSG_ID_TYPE_MAX) ||
            !strncmp(szMsgType, MSG_ID_EUC5, MSG_ID_TYPE_MAX) ||
            !strncmp(szMsgType, MSG_ID_EUB3, MSG_ID_TYPE_MAX) ||
            !strncmp(szMsgType, MSG_ID_EUB5, MSG_ID_TYPE_MAX))
        {
            ulKey = ((MSG_C_MACK *) pszMsg)->cics_trans_no.v;
            emRc = DSM_SearchKey(DC_pCBData, ulKey, &tDiskMsgHdr, &lMsgFP);
            if (emRc == DSM_MSG_OK)
            {   /* Found it, set message state in DSM to completed */
                emRc = DSM_UpdateMsgState(DC_pCBData, lMsgFP,
                                          DSM_MSG_COMPLETED);
                if (emRc != DSM_MSG_OK)
                {
```

```
                    DBCRDT_Error(DBCRDT_DSM_UPDATE_ERROR, emRc,
                                 __FILE__, __LINE__);
                }
            }
            else
            {
                sprintf(szBuffer, ">>>>>>DSM Search fail:0x%x on key 0x%x\r\n",
                        emRc, ulKey);
                DBCRDT_Display(szBuffer);
            }
            rc = 0UL;
        }
        else
        {
#endif
            memcpy(pMsgPtr, pszMsg, ulMsgLength);
            rc = DosWriteQueue(DC_RecvQ, DC_MSG_SEND, ulMsgLength,
                               pMsgPtr, ulPriority);
#ifdef DEV_SMADS
        }
#endif
    }
    else
    {
        memcpy(pMsgPtr, auchInfo, DSM_INFO_SIZE);    /* Send addressing info */
        memcpy(pMsgPtr + DSM_INFO_SIZE, pszMsg, ulMsgLength);
        rc = DosWriteQueue(DC_XmitQ, DC_MSG_SEND, ulMsgLength + DSM_INFO_SIZE,
                           pMsgPtr, ulPriority);
    }
    if (rc)
    {
        DBCRDT_Error(DBCRDT_QUEUE_WRITE_ERROR, rc, __FILE__, __LINE__);
        free(pMsgPtr);
        bRc = FALSE;
    }
    else
        bRc = TRUE;

    return bRc;
}   /* DBCRDT_SendToHost */


/*****************************************************************************
* Function:         Debit/Credit Verify Mack
* Desc:             Process the MACK from the AC.
*                   The end devices are only to recieve Positive Mack messages.
*                   It is the  responsibility of the SC to attempt to resend
*                   the message to the AC.
*                   In attempting to resend, the SC will be sending the message
*                   as it was received from the End Device.
*                   The SC is relying on the low level protocols between it and
*                   the end devices to guarantee that the message received at
*                   the SC is the same as the message sent from the end device.
*                   The SC only sends MACKs received from the AC which have a
*                   zero for the Mack Status.
*               !   The end devices are commanded to resend any unacknowledged
*               !   message once per day. It is this manner which allows an end
```

```
*               !   device to give a message to the SC multiple times.
* Inputs:           Pointer to the received MACK Message buffer, it length, and
*                   the mode it was received in.
* Outputs:          The MACK message is sent to the end device.
* Return Value:     Status of the MACK (see AC_MACK_STATUS in shapc000.h).
* External Effects: N/A
* Implementation:   N/A
*****************************************************************************/
static USHORT DBCRDT_Verify_MACK(PVOID pvExtMsg, USHORT usExtMsgLen,
                                 MSG_C_HEADER *ptMsgHdr, PUCHAR auchInfo,
                                 AC_ACCOM_MODE eMode)
{
    ULONG           ulRc, ulDatimStamp;
    LONG            lMsgFP;
    USHORT          usRc, usMackLen;
    UCHAR           szBuffer[82];
    MSG_C_MACK      sMackMsg;
    DSM_MSG_STATE   emRc;
    DSM_FILE_ERRORS efRc;

    if (usExtMsgLen > MSG_MAX_LENGTH)   /* Message to big, Nack It */
        return DC_MACK_ERROR;

    usMackLen = usExtMsgLen;
    ulRc = LIB_msg_rx_mack(pvExtMsg, &sMackMsg, &usMackLen);
    if (ulRc != LIB_SUCCESS)
        return DC_MACK_ERROR;

    /* Determine what to do next:                                    */
    /*  1. OK - Return and send next message                         */
    /*  2. RETRY - Resend message x times (see EC17) then abort      */
    /*  3. DELAY_XMIT - Disable Transmit and retry later             */
    /*  4. RESTART - Restart complete message, sequence #1           */
    /*  5. ABORT - Abort, do not resend this message                 */
    switch (sMackMsg.status)
    {
        case MSG_ACKSTS_OK:             /* Message received, processed OK */
            usRc = AC_MACK_OK;
            break;
        case MSG_ACKSTS_LENERR:         /* Message length error, retry */
            usRc = AC_MACK_RETRY;
            break;
        case MSG_ACKSTS_BUSY:           /* Received, cannot process, */
            usRc = AC_MACK_DELAY_XMIT;  /* Disable transmit          */
            DBCRDT_ACComsDisable();
            break;
        case MSG_ACKSTS_PARAMINV:       /* Invalid parameter data, retry */
            usRc = AC_MACK_RETRY;
            break;
        case MSG_ACKSTS_TRANSIDINV:     /* Invalid transaction ID, retry */
            usRc = AC_MACK_RETRY;
            break;
        case MSG_ACKSTS_TRANSREVINV:    /* Invalid revision level, retry */
            usRc = AC_MACK_RETRY;
            break;
        case MSG_ACKSTS_NOEXECUTE:      /* Unable to execute command, retry */
            usRc = AC_MACK_RETRY;
```

```
            break;
        case MSG_ACKSTS_RESTART:        /* Abort and re-start, retry */
            usRc = AC_MACK_RETRY;
            break;
        case MSG_ACKSTS_ABORT:          /* Abort..., no retry */
            usRc = AC_MACK_ABORT;
            break;
        default:
            usRc = AC_MACK_RETRY;
            break;
    }

    /* If the mode is AC_XMIT_MODE and the MACK status is AC_MACK_OK */
    /* then send the MACK to the end device.  If the message is sent */
    /* successfully then remove (complete) the message from the DSM. */
    if (eMode == AC_XMIT_MODE)
    {   /* Message is MACK status zero and EUC4, EUC5, or EUC3 */
        if ((usRc == AC_MACK_OK) &&
            (!strncmp(sMackMsg.type, MSG_ID_EUC3, MSG_ID_TYPE_MAX) ||
             !strncmp(sMackMsg.type, MSG_ID_EUC4, MSG_ID_TYPE_MAX) ||
             !strncmp(sMackMsg.type, MSG_ID_EUC5, MSG_ID_TYPE_MAX) ||
             !strncmp(sMackMsg.type, MSG_ID_EUB3, MSG_ID_TYPE_MAX) ||
             !strncmp(sMackMsg.type, MSG_ID_EUB5, MSG_ID_TYPE_MAX)))
        {   /* Complete EUC3, EUC4, EUC5, EUB3, or EUB5 message in DSM */
            emRc = DSM_ChngMsgState(DC_pCBData,
                                    ((MSG_C_MACK *) pvExtMsg)->cics_trans_no.v,
                                     DSM_MSG_COMPLETED);
            if (emRc != DSM_MSG_OK)
            {
                DBCRDT_Error(DBCRDT_DSM_UPDATE_ERROR, emRc,
                             __FILE__, __LINE__);
                sprintf(szBuffer, RED_BACK "Tx MACK %4.4s, Can NOT change "
                        "state to COMPLETED, Key=%08x,%08x." BLACK_BACK "\r\n",
                        sMackMsg.type,
                        ((MSG_C_MACK *) pvExtMsg)->cics_trans_no.v,
                        sMackMsg.cics_trans_no.v);
                DBCRDT_Display(szBuffer);
            }
            else if (!DBCRDT_SendToDevice((PUCHAR) pvExtMsg,
                                          usExtMsgLen, auchInfo))
            {   /* If MACK not sent to end device then store to send later */
                ulDatimStamp = SHTIM_time();
                efRc = DSM_MsgSave(DC_pCBData, (PUCHAR) pvExtMsg,
                                   DSM_MSG_NONPRIORITY, auchInfo,
                                   ((MSG_C_MACK *) pvExtMsg)->cics_trans_no.v,
                                    (SHORT) usExtMsgLen, ulDatimStamp, &lMsgFP);
                if (efRc != DSM_FILE_OK)
                {
                    DBCRDT_Error(DBCRDT_DSM_SAVE_ERROR, efRc,
                                 __FILE__, __LINE__);
                }
            }
        }
    }
    ptMsgHdr = ptMsgHdr;
    return usRc;
}   /* DBCRDT_Verify_MACK */
```

```
/*****************************************************************************
* Function:         Debit/Credit Transmit Message
* Desc:             Transmit Debit/Credit message to host computer.
* Inputs:           N/A
* Outputs:          N/A
* Return Value:     N/A
* External Effects: N/A
* Implementation:   N/A
*****************************************************************************/
USHORT DBCRDT_ACTransmit(PUCHAR szExtMsg, USHORT usExtMsgLength)
{
    ULONG           ulRc;
    USHORT          usErr, usMack, usMackLen;
    USHORT          usOrigMsgLength, usTransmitRecvLength;
    BOOL            bPolling;
    UCHAR           szBuffer[82], auchInfo[DSM_INFO_SIZE];
    MSG_C_MACK      sMackMsg;
    MSG_C_HEADER    tMsgHdr;
#if FALSE
    DSM_MSG_STATE   emRc;
#endif

    /* Display transmit message header */
    ulRc = LIB_msg_get_header(&szExtMsg[DSM_INFO_SIZE], &tMsgHdr);
    if (ulRc == LIB_SUCCESS)
    {
        sprintf(szBuffer, BROWN_BACK "Tx %4.4s Len 0x%x" BLACK_BACK "\r\n",
                tMsgHdr.type, usExtMsgLength);
    }
    else    /* NOTE: No MACKs should go up the R-line */
    {
        tMsgHdr.type[0] = 0;
        usMackLen = usExtMsgLength - DSM_INFO_SIZE;
        ulRc = LIB_msg_rx_mack(&szExtMsg[DSM_INFO_SIZE], &sMackMsg,
                               &usMackLen);
        if (ulRc == LIB_SUCCESS)
        {
            sprintf(szBuffer, BROWN_BACK "Tx MACK %4.4s Status %d" BLACK_BACK
                    "\r\n", sMackMsg.type, sMackMsg.status);
        }
        else
        {
            strcpy(szBuffer, "DBCRDT_ACTransmit: Unknown message! ***\r\n");
        }
    }
    DBCRDT_Display(szBuffer);
    /* End display transmit message header */

    if (DBCRDT_ACComsDisabled() &&
        strncmp(&szExtMsg[DSM_INFO_SIZE], MSG_ID_EUC0, MSG_ID_TYPE_MAX))
    {   /* If comms disabled and this is not an EUC0 */
        usMack = AC_MACK_DELAY_XMIT;
    }
    else
    {
```

```
bPolling = FALSE;
usOrigMsgLength = usExtMsgLength - DSM_INFO_SIZE;
memcpy(auchInfo, szExtMsg, DSM_INFO_SIZE);
memcpy(pDC_sExtMsg, &szExtMsg[DSM_INFO_SIZE], usOrigMsgLength);
usErr = SH_DCXmit((PCHAR) pDC_sExtMsg, &usOrigMsgLength,
                  (PCHAR) pDC_sRcvdMsg, &usTransmitRecvLength,
                  bPolling);
if ((usErr == AC_ERROR_CONTINUE) || (usTransmitRecvLength == 0))
{
    DC_eCommsStatus = DBCRDT_COMMS_DOWN;
    if (tMsgHdr.type[0])
    {
        sprintf(szBuffer, BLUE_BROWN "DBCRDT_ACTransmit: Send %4.4s "
                "FAILED." WHITE_BLACK "\r\n", tMsgHdr.type);
    }
    else
    {
        sprintf(szBuffer, BLUE_BROWN "DBCRDT_ACTransmit: Send MACK "
                "%4.4s FAILED." WHITE_BLACK "\r\n", sMackMsg.type);
    }
    DBCRDT_Display(szBuffer);
    usMack = AC_MACK_ABORT;
}
else if (usErr == AC_ERROR_OK)
{
    usMack = DBCRDT_Verify_MACK(pDC_sRcvdMsg, usTransmitRecvLength,
                                &tMsgHdr, auchInfo, AC_XMIT_MODE);
    if (usMack == AC_MACK_DELAY_XMIT)
    {
        DBCRDT_ACComsDisable();
    }
    else if (usMack == DC_MACK_ERROR)
    {
        usMack = AC_MACK_ABORT;
    }
    else
    {   /* If no AC error then DBCRDT_Verify_MACK was called, and it */
        /* changed the status of the message on Disk.  If an error   */
        /* is encountered then the message status is not modified.   */
        DBCRDT_ACComsEnable();  /* If it was disabled, reenable it */

        /* Display received message header */
        usMackLen = usTransmitRecvLength;
        ulRc = LIB_msg_rx_mack((PCHAR) pDC_sRcvdMsg, &sMackMsg,
                               &usMackLen);
        if (ulRc == LIB_SUCCESS)
        {
            sprintf(szBuffer,
                    BLUE_BROWN "Tx MACK %4.4s Status %d" WHITE_BLACK
                    "\r\n", sMackMsg.type, sMackMsg.status);
        }
        else
        {
            strcpy(szBuffer,
                   "DBCRDT_ACTransmit: Unknown message rcvd! ***\r\n");
        }
        DBCRDT_Display(szBuffer);
```

```
                /* End display received message header */
                if (!strncmp(sMackMsg.type, MSG_ID_EUC0, MSG_ID_TYPE_MAX))
                {
                    DC_eCommsStatus = DBCRDT_COMMS_R_LINE_UP;
                }
                else
                {
                    DC_eCommsStatus = DBCRDT_COMMS_UP;
                }
            }
        }
        else    /* Must be AC_ERROR_ABORT, AC_ERROR_RESET, or   */
        {       /* AC_ERROR_SHUTDOWN return from SH_DCXmit call */
            DC_eCommsStatus = DBCRDT_COMMS_DOWN;
            if (tMsgHdr.type[0])
            {
                sprintf(szBuffer, BLUE_BROWN "DBCRDT_ACTransmit: SH_DCXmit "
                        "%4.4s error, %d." WHITE_BLACK "\r\n", tMsgHdr.type,
                        usErr);
            }
            else
            {
                sprintf(szBuffer, BLUE_BROWN "DBCRDT_ACTransmit: SH_DCXmit "
                        "MACK %4.4s error, %d." WHITE_BLACK "\r\n",
                        sMackMsg.type, usErr);
            }
            DBCRDT_Display(szBuffer);
            usMack = AC_MACK_ABORT;
        }
    }
    return usMack;
}   /* DBCRDT_ACTransmit */


/*****************************************************************************
* Function:         Debit/Credit Transmit Messages
* Desc:             Remove Debit/Credit messages from transmit queue and
*                   transmit them to the host computer.  This thread handles
*                   EUC1, EUC3, EUC4, EUC5, EUC6, MACKEUC1, MACKEUC3, MACKEUC4,
*                   MACKEUC5, and MACKEUC6 messages only.
* Inputs:           N/A
* Outputs:          N/A
* Return Value:     N/A
* External Effects: N/A
* Implementation:   N/A
*****************************************************************************/
VOID ENV_CDECL DBCRDT_XmitMsgs(PVOID dummy)
{
    APIRET          rc;
    ULONG           ulExtMsgLength, ulMsgID;
    BOOL            bRunning;
    UCHAR           szExtMsg[MSG_MAX_LENGTH];

    /* Local Initialization */
    bRunning = TRUE;

    /* Post 'Initialized' semaphore */
```

```
    rc = DosPostEventSem(DC_InitSem);
    if (rc && (rc != ERROR_ALREADY_POSTED))
    {
        DBCRDT_Error(DBCRDT_SEM_POST_ERROR, rc, __FILE__, __LINE__);
        bRunning = FALSE;
    }

    while (bRunning)
    {
        if (DBCRDT_GetQueueMsg(DC_XmitQ, szExtMsg, &ulExtMsgLength, &ulMsgID))
        {
            if (ulMsgID == DC_MSG_EXIT)
            {
                bRunning = FALSE;
            }
            else if (ulExtMsgLength == 0UL)
            {
                DBCRDT_Error(DBCRDT_QUEUE_EMPTY_MSG, rc, __FILE__, __LINE__);
            }
            else if (ulMsgID == DC_MSG_SEND)
            {
                DBCRDT_ACTransmit(szExtMsg, (USHORT) ulExtMsgLength);
            }
            else
            {
                DBCRDT_Display("DBCRDT_XmitMsgs: Invalid queue message.\r\n");
            }
        }
        else
        {
            DosSleep(50L);              /* Relinquish CPU cycles */
        }
    }
    DC_iXmitMsgsId = -1;

    /* Post 'Closing' semaphore */
    rc = DosPostEventSem(DC_XmitEndSem);
    if (rc && (rc != ERROR_ALREADY_POSTED))
    {
        DBCRDT_Error(DBCRDT_SEM_POST_ERROR, rc, __FILE__, __LINE__);
    }
    dummy = dummy;
    _endthread();
}   /* DBCRDT_XmitMsgs */


/*****************************************************************************
* Function:         Debit/Credit Manage DSM Messages
* Desc:             Resend DSM messages that have not been sent yet.
* Inputs:           N/A
* Outputs:          N/A
* Return Value:     N/A
* External Effects: N/A
* Implementation:   N/A
*****************************************************************************/
VOID ENV_CDECL DBCRDT_ManageDSM(PVOID dummy)
{
```

```
    APIRET          rc;
    INT             iLcv, iResend;
    ULONG           ulCurTime, ulPriority, ulMsgsInQ;
    LONG            lStartPtr, lMsgAdr;
    SHORT           sLength;
    UCHAR           szExtMsg[MSG_MAX_LENGTH];
    PUCHAR          pMsgPtr;
    LONGINT         tMsgType;
    DSM_CBUF_HDR    tDiskMsgHdr;
    DSM_MSG_STATE   eRc, emRc;

    /* Post 'Initialized' semaphore */
    rc = DosPostEventSem(DC_InitSem);
    if (rc && (rc != ERROR_ALREADY_POSTED))
    {
        DC_bRunning = FALSE;
        DBCRDT_Error(DBCRDT_SEM_POST_ERROR, rc, __FILE__, __LINE__);
    }
#ifdef DEV_SMADS
    iResend = DC_iSecondsForResend;     /* Variable Seconds */
    for (iLcv = 0; iLcv < 1800; iLcv++)
    {   /* Wait 15 minutes before starting */
        DosSleep(500L);
        if (!DC_bRunning)
        {
            break;
        }
    }
#else
    iResend = 900;                                          /* 15 Minutes in
seconds */
#endif

    while (DC_bRunning)
    {
        rc = DosQueryQueue(DC_XmitQ, &ulMsgsInQ);
        if (rc)
        {
            DBCRDT_Error(DBCRDT_QUEUE_QUERY_ERROR, rc, __FILE__, __LINE__);
            ulMsgsInQ = ULONG_MAX;
        }
        ulCurTime = SHTIM_time();
        lStartPtr = -1L;
        while (DC_bRunning && (ulMsgsInQ < 5UL))
        {   /* Check for unsent non-priority messages */
            eRc = DSM_MsgGetPtr(DC_pCBData, &tDiskMsgHdr, szExtMsg, &sLength,
                                DSM_MSG_NONPRIORITY | DSM_MSG_SENT, &lMsgAdr,
                                &lStartPtr);
            if (eRc == DSM_MSG_OK)
            {
                if (tDiskMsgHdr.DatimStamp < ulCurTime - iResend)
                {   /* If more than specified age */
                    LIB_cnv_ebcdic_to_ascii(szExtMsg, tMsgType.b,
                                            MSG_ID_TYPE_MAX);
                    if (!strncmp(tMsgType.b, MSG_ID_MACK, MSG_ID_TYPE_MAX))
                    {   /* Send MACK EUC3, EUC4, or EUC5 to AVM */
                        if (DBCRDT_SendToDevice(szExtMsg, (USHORT) sLength,
```

```
                                        tDiskMsgHdr.ucInfo))
                    {   /* If MACK sent to end device then delete MACK */
                        emRc = DSM_UpdateMsgState(DC_pCBData, lMsgAdr,
                                                  DSM_MSG_COMPLETED);
                        if (emRc != DSM_MSG_OK)
                            DBCRDT_Error(DBCRDT_DSM_UPDATE_ERROR, emRc,
                                         __FILE__, __LINE__);
                    }
#ifdef DEV_SMADS
                    else
                    {   /* For TRANSIT AUTHORITY, wait for device to act */
                        /* on last message before continuing */
                        DosSleep(2500L);
                    }
#endif
                }
                else if ((!strncmp(tMsgType.b, MSG_ID_EUC0,
                                   MSG_ID_TYPE_MAX - 1))
                                     ||
                         (!strncmp(tMsgType.b, MSG_ID_EUB2,
                                   MSG_ID_TYPE_MAX - 1)))
                {   /* Send EUC3, EUC4, or EUC5 to AC */
                    if (!DBCRDT_ACComsDisabled())
                    {
                        pMsgPtr = (PUCHAR) malloc(sLength + DSM_INFO_SIZE);
                        if (pMsgPtr == NULL)
                            DBCRDT_Error(DBCRDT_MEMORY_ERROR, 0UL,
                                         __FILE__, __LINE__);
                        else
                        {   /* Place message on D/C transmit queue */
                            ulPriority = 0;
                            memcpy(pMsgPtr, tDiskMsgHdr.ucInfo,
                                   DSM_INFO_SIZE);
                            memcpy(pMsgPtr + DSM_INFO_SIZE, szExtMsg,
                                   sLength);
                            rc = DosWriteQueue(DC_XmitQ, DC_MSG_SEND,
                                               (ULONG) (sLength +
                                                        DSM_INFO_SIZE),
                                               pMsgPtr, ulPriority);
                            if (rc)
                            {
                                DBCRDT_Error(DBCRDT_QUEUE_WRITE_ERROR, rc,
                                             __FILE__, __LINE__);
                                free(pMsgPtr);
                            }
#ifdef DEV_SMADS
                            /* Wait one second to avoid filling up queue */
                            /* to fast.  CC response is not too quick    */
                            DosSleep(1000L);
#endif
                        }
                    }
                }
                else
                {
                    DBCRDT_Error(DBCRDT_DSM_MSG_ERROR, tMsgType.v,
                                 __FILE__, __LINE__);
```

```
                    }
                }
                else
                    break;
            }
            else
                break;
        }
        for (iLcv = 0; iLcv < 1800; iLcv++)
        {                                   /* Check every 15 minutes */
            DosSleep(500L);
            if (!DC_bRunning)
                break;
        }
    }
    DC_iManagDSMId = -1;

    /* Post 'Closing' semaphore */
    rc = DosPostEventSem(DC_DSMEndSem);
    if (rc && (rc != ERROR_ALREADY_POSTED))
    {
        DBCRDT_Error(DBCRDT_SEM_POST_ERROR, rc, __FILE__, __LINE__);
    }
    dummy = dummy;
    _endthread();
}   /* DBCRDT_ManageDSM */


/*****************************************************************************
* Module:       Get_LIB_Result
* Desc:         Examine the long returned by a call to table manager library
*               and return the approriate MACK status.
* Inputs:       lrc - Long integer to be examined.
* Outputs:      Status - Char field that is used for Mack Status.
* Errors:       None.
*****************************************************************************/
static UCHAR DBCRDT_GetLIBResult(ULONG ulRc)
{
    UCHAR    ucStatus;

    if (ulRc == LIB_SUCCESS)
        ucStatus = MSG_ACKSTS_OK;
    else if (ulRc == LIB_W_MSG_INVVER)
        ucStatus = MSG_ACKSTS_TRANSREVINV;
    else if (ulRc == LIB_W_MSG_UNKNOWN || ulRc == LIB_E_MSG_NULLPTR)
        ucStatus =  MSG_ACKSTS_TRANSIDINV;
    else if (ulRc == LIB_E_MSG_INVLEN)
        ucStatus =  MSG_ACKSTS_LENERR;
    else
        ucStatus = MSG_ACKSTS_PARAMINV;
    return ucStatus;
}


/*****************************************************************************
* Module:       ProcessEC03Msg
* Desc:         Message EC03 is the Delay Transmit Message.  The DC_ACXmitSem
```

```
*               semaphore is set to the Disabled state if this command informs
*               us that communications with the AC Debit/Credit is disabled.
* Inputs:       EC03 Message.
* Outputs:      ACXmitSem
* Errors:       Default error handling.
*****************************************************************************/
static UCHAR DBCRDT_ProcessEC03Msg(MSG_EC03 *psEC03Msg)
{
    UCHAR   ucStatus, ucEC03Value;

    /* Get the new communications value */
    ucEC03Value = psEC03Msg->fdata.xmit_ctrl;
    if ((ucEC03Value != 0) && (ucEC03Value != 1))
        ucStatus = MSG_ACKSTS_PARAMINV;
    else
    {   /* Store the new value in to Table Manger ER63 structure */
        ucStatus = MSG_ACKSTS_OK;
        if (ucEC03Value == 1)
            DBCRDT_ACComsDisable();
    }
    return ucStatus;
}
```

□

APPENDIX C

```
 IDENTIFICATION DIVISION.                                                00000010
 PROGRAM-ID.  CWRB7100.                                                  00000020
 AUTHOR.      CUBIC/CARCG                                                00000030
 INSTALLATION.                                                           00000040
 DATE-WRITTEN.  JANUARY 2000.                                            00000050
 DATE-COMPILED.                                                          00000060
*                                                                        00000070
************************************************                        00000080
**   PROGRAM NAME:  MONTHLY BENEFITS / CLAIMS ACTIVITY REPORT           00000090
**   PROGRAM ID:    CWRB7100                                            00000091
**   SYSTEM:  9121-490, MVS/XA, CICS,COBOL II, ORACLE                   00000092
**   PROJECT: ELECTRONIC BENEFITS DISTRIBUTION SYSTEM                   00000093
**                                                                      00000094
**   DESC:                                                              00000095
**                                                                      00000096
**                                                                      00000097
**                                                                      00000098
**                                                                      00000099
**   INPUTS:                                                            00000100
**       REFERENCE FILE      VSAM RANDOM ACCESS FILE                    00000110
**       BENEFITS TABLE      ORACLE FILE #                              00000120
**       CLAIMS TABLE        ORACLE FILE #                              00000130
**       DEFINITION TABLE    ORACLE FILE #                              00000140
**       CUSTOMER TABLE      ORACLE FILE #                              00000150
**       CARDHOLDER TABLE    ORACLE FILE #                              00000160
**                                                                      00000170
**   OUTPUTS:                                                           00000180
**       MONTHLY BENEFITS / CLAIMS ACTIVITY REPORT                      00000190
**                                                                      00000191
**   PROGRAM CALLS:                                                     00000192
**       NONE                                                           00000193
**                                                                      00000194
**   REVISION HISTORY / REMARKS                                         00000195
**     01/00   SMB000  HYNSON  INIT CODING                              00000196
**                                                                      00000197
************************************************************************00000198
/                                                                       00000199
 ENVIRONMENT DIVISION.                                                  00000200
 CONFIGURATION SECTION.                                                 00000210
 SPECIAL-NAMES.  C01 IS TOP-OF-PAGE.                                    00000220
 SOURCE-COMPUTER.  ES-9000.                                             00000230
 OBJECT-COMPUTER.  ES-9000.                                             00000240
*                                                                       00000250
 INPUT-OUTPUT SECTION.                                                  00000260
 FILE-CONTROL.                                                          00000270
************************************************************************00000280
*                                                                      *00000290
* COPYLIB NAME       :  REFER SELECT/ASSIGN                            *00000291
* COPYLIB ID         :  CWAL0300                                       *00000292
*                                                                      *00000293
* SYSTEM:  9121-490, MVS/XA,  CICS, COBOL II, VSAM                     *00000294
* PROJECT: 170-2660, ELECTRONIC BENEFITS DISTRIBUTION SYSTEM           *00000295
*                    ON-LINE MONITORING AND CONTROL                    *00000296
*                                                                      *00000297
* DESC:     THIS COPYLIB CONTAINS THE SELECT/ASSIGN STATEMENT          *00000298
*           FOR THE REFERENCE TABLE FILE.                              *00000299
*                                                                      *00000300
```

```
*****************************************************************00000310
* REVISION HISTORY:                                             *00000320
*                                                               *00000330
* 04/10/97        JLK  INITIAL CODING                           *00000340
*****************************************************************00000350
                                                                 00000360
     SELECT REFER-FILE                    ASSIGN TO DEFMF03      00000370
            ORGANIZATION INDEXED                                 00000380
            ACCESS       IS DYNAMIC                              00000390
            RECORD KEY   IS RF-PRIME-KEY                         00000391
            FILE STATUS  IS RF-STATUS.                           00000392
                                                                 00000393
     SELECT RPT-FILE   ASSIGN TO RPTFILE.                        00000394
*                                                                00000395
 DATA DIVISION.                                                  00000396
 FILE SECTION.                                                   00000397
*                                                                00000398
*****************************************************************00000399
*                                                               *00000400
* COPYLIB NAME       :  REFER FILE DESCRIPTION                  *00000410
* COPYLIB ID         :  CWDL0300                                *00000420
*                                                               *00000430
* SYSTEM:  9121-490, MVS/XA,  CICS, COBOL II, VSAM              *00000440
* PROJECT: 170-2660, ELECTRONIC BENEFITS DISTRIBUTION SYSTEM    *00000450
*                    ON-LINE MONITORING AND CONTROL             *00000460
*                                                               *00000470
* DESC:     THIS COPYLIB CONTAINS THE FILE DESCRIPTION          *00000480
*           FOR THE REFERENCE TABLE FILE.                       *00000490
*                                                               *00000491
*****************************************************************00000492
* REVISION HISTORY:                                             *00000493
*                                                               *00000494
* 01/25/99        rdrk INITIAL CODING                           *00000495
*****************************************************************00000496
                                                                 00000497
 FD  REFER-FILE.                                                 00000498
 01  RF-RECORD.                                                  00000499
     05  RF-PRIME-KEY               PIC X(17).                   00000500
     05  RF-DATA-ELEMENTS.                                       00000510
**   START-EXPIRATION-DATE FORMAT YYYYMMDD                       00000520
         10  RF-START-EXPIRATION-DATE    PIC  X(08).            00000530
         10  FILLER                      PIC  X(01).            00000540
**   END-EXPIRATION-DATE FORMAT YYYYMMDD                         00000550
         10  RF-END-EXPIRATION-DATE      PIC  X(08).            00000560
         10  FILLER                      PIC  X(39).            00000570
         10  RF-ENTRY-STATUS             PIC  X(01).            00000580
             88  RF-ENTRY-ACTIVE         VALUE 'A'.             00000590
             88  RF-ENTRY-INACTIVE       VALUE 'I'.             00000591
*                                                                00000592
*                                                                00000593
 FD  RPT-FILE                                                    00000594
     RECORDING MODE IS F.                                        00000595
 01  RPT-REC          PIC X(132).                               00000596
*                                                                00000597
/                                                                00000598
 WORKING-STORAGE SECTION.                                        00000599
 01  FILLER     PIC X(30)  VALUE 'WORKING STORAGE STARTS HERE'.  00000600
```

```
      *                                                                 00000610
      ************************************************                  00000620
      **                                            **                  00000630
      **           SQLCODE   PARAMETERS             **                  00000640
      **                                            **                  00000650
      ************************************************                  00000660
      **                                                                00000670
      ****************************************************************00000680
      *                                                               *00000690
      * COPYLIB NAME  : ORACLE SQLCODES                               *00000691
      * COPYLIB ID    : CWELB000                                      *00000692
      *                                                               *00000693
      * SYSTEM:  9121-490  MVS/ESA, CICS, COBOL II, ORACLE            *00000694
      * PROJECT: 170-2719 ELECTRONIC BENEFITS DISTRIBUTION SYSTEM     *00000695
      *                                                               *00000696
      ****************************************************************00000697
      * REVISION HISTORY :                                            *00000698
      *                                                               *00000699
      * 01/05/00 SMB000 RONO INITIAL                                  *00000700
      ****************************************************************00000710
      ****************************************************************00000720
       01  ORACLE-SQL-CODES.                                           00000730
           05  ORA-NAMED-SQLCODE              PIC S9(8) COMP.          00000740
               88  ORA-SQL-SUCCESSFUL             VALUE 0.             00000750
             88  ORA-SQL-WARNING                VALUE +001 THRU +9999. 00000760
               88  ORA-SQL-ROW-NOT-FOUND          VALUE +1403, +100.   00000770
               88  ORA-SQL-END-OF-FETCH           VALUE +1403, +100.   00000780
               88  ORA-SQL-GENERAL-ERROR          VALUE -9999 THRU -001. 00000790
               88  ORA-SQL-DUPLICATE-ROW          VALUE -1.            00000791
               88  ORA-SQL-NOT-LOGGED-ON          VALUE -1012.         00000792
               88  ORA-SQL-INVALID-COLUMN         VALUE -904.          00000793
           05  ORA-SQLCODE-DISP-4             PIC  -(4)9.              00000794
           05  ORA-SQLCODE-DISP-8             PIC  -(8)9.              00000795
           05  ORA-TABLE-ID                   PIC  X(20).              00000796
           05  ORA-FUNCTION-ID                PIC  X(08).              00000797
      /                                                                 00000798
       01  PROGRAM-STATUSES.                                           00000799
           05  RF-STATUS                 PIC X(02) VALUE SPACES.       00000800
               88  VALID-STATUS              VALUE '00', '97'.         00000810
      **                                                                00000820
      ************************************************                  00000830
      **        HOST VARIABLE DEFINITIONS           **                  00000840
      ************************************************                  00000850
      **                                                                00000860
           EXEC SQL  BEGIN DECLARE SECTION  END-EXEC.                   00000870
      **                                                                00000880
      ************************************************                  00000890
      **           BENEFITS   DATA    DESCRIPTIONS   **                  00000891
      **                                                                00000892
      ****************************************************************00000893
      *                                                                 00000894
-INC CWELB100                                                           0000089%
      *                                                                 00000896
           EXEC SQL VAR                                                 00000897
             BEN-LOAD-DT-TM          IS  DATE                           00000898
           END-EXEC                                                     00000899
                                                                        00000900
```

```
           EXEC SQL VAR                                                  00000950
             BEN-EXPIRATION-DATE     IS  DATE                          00000960
           END-EXEC                                                      00000970
                                                                         00000980
           EXEC SQL VAR                                                  00000990
             BEN-LAST-CLAIM-DT-TM    IS  DATE                          00000991
           END-EXEC                                                      00000992
                                                                         00000993
           EXEC SQL VAR                                                  00000994
             BEN-LAST-REQUEST-DT-TM  IS  DATE                          00000995
           END-EXEC                                                      00000996
                                                                         00000997
           EXEC SQL VAR                                                  00000998
             BEN-HOLD-DT-TM          IS  DATE                          00000999
           END-EXEC                                                      00001000
                                                                         00001010
           EXEC SQL VAR                                                  00001020
             BEN-UPDATE-DT-TM        IS  DATE                          00001030
           END-EXEC                                                      00001040
                                                                         00001050
           EXEC SQL VAR                                                  00001060
             BEN-INITIAL-VAL-AMT     IS DECIMAL(5,2)                   00001070
           END-EXEC                                                      00001080
                                                                         00001090
           EXEC SQL VAR                                                  00001091
             BEN-REM-VAL-AMT         IS DECIMAL(5,2)                   00001092
           END-EXEC                                                      00001093
                                                                         00001094
           EXEC SQL VAR                                                  00001095
             BEN-LAST-CLAIM-VAL-AMT  IS DECIMAL(5,2)                   00001096
           END-EXEC                                                      00001097
      /                                                                  00001098
      ************************************************                   00001099
      **           CLAIMS    DATA     DESCRIPTIONS      **              00001100
      **                                                                00001110
      ********************************************************************00001120
-INC CWELB500                                                            0000113%
*                                                                        00001140
           EXEC SQL VAR                                                  00001150
             CLM-EFFECTIVE-DATE      IS  DATE                          00001160
           END-EXEC                                                      00001170
                                                                         00001180
           EXEC SQL VAR                                                  00001190
             CLM-EXPIRATION-DATE     IS  DATE                          00001191
           END-EXEC                                                      00001192
                                                                         00001193
           EXEC SQL VAR                                                  00001198
             CLM-CLAIM-VAL-AMT       IS DECIMAL(5,2)                   00001199
           END-EXEC                                                      00001200
      **                                                                 00001210
      /                                                                  00001220
      ************************************************                   00001230
      **           DEFINITIONS  DATA  DESCRIPTIONS      **              00001240
      **                                                                00001250
       01  DEFINITION-ROW.                                              00001260
           05  DEF-BENEFIT-DESC       PIC X(60).                        00001270
           05  DEF-BENEFIT-TYPE       PIC X(05).                        00001280
```

```
**                                                              00001290
**                                                              00001291
**************************************************              00001292
**          CUSTOMER  DATA  DESCRIPTIONS          **            00001293
**                                                              00001294
 01  CUSTOMER-ROW.                                              00001295
     05  CUST-CUSTOMER-NAME    PIC X(50).                       00001296
     05  CUST-STREET-ADDR1     PIC X(40).                       00001297
     05  CUST-STREET-ADDR2     PIC X(40).                       00001298
     05  CUST-CITY             PIC X(15).                       00001299
     05  CUST-STATE            PIC X(02).                       00001300
     05  CUST-ZIPCODE          PIC X(05).                       00001310
**                                                              00001320
**                                                              00001330
**************************************************              00001340
**         CARDHOLDER  DATA  DESCRIPTIONS         **            00001350
**                                                **            00001360
 01  CARDHOLDER-ROW.                                            00001370
     05  CRDH-LAST-NAME        PIC X(15).                       00001380
     05  CRDH-FIRST-NAME       PIC X(15).                       00001390
**                                                              00001391
**                                                              00001392
**                                                              00001393
**************************************************              00001394
**          HOST  VARIABLE  DESCRIPTIONS          **            00001395
**                                                **            00001396
 01  WS-HOST-SEARCH-VARIABLES      VALUE SPACES.                00001397
     05  WS-START-EXPIRATION-DATE  PIC X(09).                   00001398
     05  WS-END-EXPIRATION-DATE    PIC X(09).                   00001399
**                                                              00001400
     05  WS-OLD-CUSTOMER-ID                PIC X(14).           00001410
     05  WS-OLD-SERIAL-NUMBER              PIC X(15).           00001420
     05  WS-OLD-MFG-SERIAL-NUMBER          PIC X(11).           00001430
     05  WS-CURRENT-DATE                   PIC X(14).           00001440
/                                                               00001450
**************************************************              00001460
**                                                **            00001470
**      BENEFITS TABLE DECLARATION                **            00001480
**                                                **            00001490
**************************************************              00001491
**                                                              00001492
      EXEC SQL                                                  00001493
          DECLARE BENE CURSOR FOR                               00001494
              SELECT TO_CHAR(EFFECTIVE_DATE, 'MM/DD/YY'),       00001495
                     SERIAL_NUM,                                00001496
                     CUSTOMER_ID,                               00001497
                     BENEFIT_TYPE,                              00001498
                     EXPIRATION_DATE,                           00001499
                     MFG_SERIAL_NUM,                            00001500
                     INITIAL_VAL_AMT                            00001510
              FROM   MCHECK.BENEFITS                            00001520
             WHERE  EXPIRATION_DATE >= :WS-START-EXPIRATION-DATE 00001530
               AND  EXPIRATION_DATE <= :WS-END-EXPIRATION-DATE  00001540
              ORDER BY                                          00001550
                     CUSTOMER_ID,                               00001560
                     SERIAL_NUM,                                00001570
                     EXPIRATION_DATE,                           00001580
```

```
                    EFFECTIVE_DATE                                      00001590
      END-EXEC.                                                         00001591
/                                                                       00001592
**************************************************                      00001593
**                                              **                      00001594
**        CLAIMS TABLE DECLARATION              **                      00001595
**                                              **                      00001596
**************************************************                      00001597
**                                                                      00001598
     EXEC SQL                                                           00001599
         DECLARE CLMS CURSOR FOR                                        00001600
             SELECT TO_CHAR(REQUEST_DT_TM, 'MM/DD/YY'),                 00001610
                    EXPIRATION_DATE,                                    00001620
                    SERIAL_NUM,                                         00001630
                    MFG_SERIAL_NUM,                                     00001640
                    BENEFIT_TYPE,                                       00001650
                    CLAIM_VAL_AMT                                       00001660
             FROM   MCHECK.CLAIMS                                       00001670
             WHERE  EXPIRATION_DATE >= :WS-START-EXPIRATION-DATE        00001680
               AND  EXPIRATION_DATE <= :WS-END-EXPIRATION-DATE          00001690
               AND  SERIAL_NUM      = :WS-OLD-SERIAL-NUMBER             00001691
               AND  MFG_SERIAL_NUM = :WS-OLD-MFG-SERIAL-NUMBER          00001692
           ORDER BY EXPIRATION_DATE,                                    00001693
                    REQUEST_DT_TM                                       00001694
     END-EXEC.                                                          00001695
*                                                                       00001696
     EXEC SQL END DECLARE SECTION  END-EXEC.                            00001697
/                                                                       00001698
**************************************************                      00001699
**                                              **                      00001700
**       ORACLE  SQL COMMUNICATION AREA         **                      00001710
**                                              **                      00001720
**************************************************                      00001730
**                                                                      00001740
     EXEC SQL    INCLUDE SQLCA     END-EXEC.                            00001750
                                                                        00001760
 01  MSG-TEXT    PIC X(200).                                           00001770
 01  MAX-SIZE    PIC S9(9) COMP VALUE 200.                             00001780
 01  MSG-LENGTH  PIC S9(9) COMP.                                       00001790
/                                                                       00001791
**************************************************                      00001792
**                                              **                      00001793
**       REPORT   DEFINITION   AREA             **                      00001794
**                                              **                      00001795
**************************************************                      00001796
**                                                                      00001797
 01  HEADING-LINE1.                                                    00001798
     05  FILLER             PIC X(10)  VALUE 'RUN DATE: '.             00001799
     05  HL1-REPORT-DATE.                                              00001800
         10  HL1-RD-MONTH   PIC Z9     VALUE ZEROES.                   00001810
         10  FILLER         PIC X      VALUE '/'.                      00001820
         10  HL1-RD-DAY     PIC Z9     VALUE ZEROES.                   00001830
         10  FILLER         PIC X      VALUE '/'.                      00001840
         10  HL1-RD-YEAR    PIC X(04)  VALUE SPACES.                   00001850
     05  FILLER             PIC X(13)  VALUE SPACES.                   00001860
     05  FILLER             PIC X(23)                                  00001870
           VALUE 'WASHINGTON METROPOLITAN'.                             00001880
```

```
     05  FILLER              PIC X(23)                              00001890
           VALUE ' AREA TRANSIT AUTHORITY'.                        00001891
     05  FILLER              PIC X(30)  VALUE SPACES.               00001892
     05  FILLER              PIC X(06)  VALUE 'PAGE: '.             00001893
     05  HL1-PAGE-NUMBER     PIC ZZ9    VALUE ZEROES.               00001894
**                                                                  00001895
 01  HEADING-LINE1A.                                               00001896
     05  FILLER              PIC X(10)  VALUE 'RUN TIME: '.         00001897
     05  HL1-RUN-TIME        PIC X(08)  VALUE SPACES.               00001898
     05  FILLER              PIC X(15)  VALUE SPACES.               00001899
     05  FILLER              PIC X(46)                              00001900
         VALUE 'BENEFITS / CLAIMS ACTIVITY REPORT FOR BENEFITS'.    00001910
     05  FILLER              PIC X(29)  VALUE SPACES.               00001920
     05  FILLER              PIC X(06)  VALUE ' PGM: '.             00001930
     05  HL1-PROGRAM-NME     PIC X(08)  VALUE SPACES.               00001940
**                                                                  00001950
 01  HEADING-LINE1B.                                               00001960
     05  FILLER              PIC X(37)  VALUE SPACES.               00001970
     05  FILLER              PIC X(17)                              00001980
         VALUE 'EXPIRING  FROM   '.                                 00001990
     05  HL1-START-DATE      PIC X(09)  VALUE SPACES.               00001991
     05  FILLER              PIC X(04)  VALUE ' TO '.               00001992
     05  HL1-END-DATE        PIC X(09)  VALUE SPACES.               00001993
**                                                                  00001994
*                                                                   00001995
*************************************************************     00001996
*                 CUSTOMER INFORMATION HEADINGS                *    00001997
*************************************************************     00001998
*                                                                   00001999
 01  HEADING-LINE2.                                                00002000
     05  FILLER              PIC X(12)  VALUE SPACES.               00002010
     05  FILLER              PIC X(12)  VALUE 'CUSTOMER ID '.       00002020
     05  HL2-CUSTOMER-ID     PIC X(14)  VALUE SPACES.               00002030
     05  FILLER              PIC X(04)  VALUE SPACES.               00002040
     05  FILLER              PIC X(15)  VALUE 'CUSTOMER INFO: '.    00002050
     05  HL2-CUSTOMER-NAME   PIC X(50)  VALUE SPACES.               00002060
*                                                                   00002070
 01  HEADING-LINE3                      VALUE SPACES.              00002080
     05  FILLER              PIC X(57).                             00002090
     05  HL3-CUST-ADDR1      PIC X(40).                             00002091
*                                                                   00002092
 01  HEADING-LINE4                      VALUE SPACES.              00002093
     05  FILLER              PIC X(57).                             00002094
     05  HL4-CUST-ADDR2      PIC X(40).                             00002095
*                                                                   00002096
 01  HEADING-LINE5.                                                00002097
     05  FILLER              PIC X(57)  VALUE SPACES.               00002098
     05  HL5-CITY-STATE-ZIP  PIC X(40)  VALUE SPACES.               00002099
*                                                                   00002100
*************************************************************     00002110
*                 REPORT DETAIL INFORMATION                    *    00002120
*************************************************************     00002130
*                                                                   00002140
 01  DETAIL-HEADING.                                               00002150
     05  FILLER              PIC X(13)  VALUE SPACES.               00002160
     05  FILLER              PIC X(05)  VALUE 'EVENT'.              00002170
     05  FILLER              PIC X(28)  VALUE SPACES.               00002180
```

```
     05  FILLER              PIC X(04)  VALUE 'DATE'.              00002190
     05  FILLER              PIC X(11)  VALUE SPACES.              00002191
     05  FILLER              PIC X(05)  VALUE 'VALUE'.             00002192
     05  FILLER              PIC X(09)  VALUE SPACES.              00002193
     05  FILLER              PIC X(04)  VALUE 'AUTH'.              00002194
     05  FILLER              PIC X(08)  VALUE SPACES.              00002195
     05  FILLER              PIC X(07)  VALUE 'CLAIMED'.           00002196
     05  FILLER              PIC X(05)  VALUE SPACES.              00002197
     05  FILLER              PIC X(09)  VALUE 'UNCLAIMED'.         00002198
*                                                                  00002199
 01  DETAIL-LINE1.                                                00002200
     05  FILLER              PIC X(05)  VALUE SPACES.              00002210
     05  FILLER              PIC X(18)                             00002220
             VALUE 'CARDHOLDER NAME:  '.                          00002230
     05  DL1-CARDHOLDER-NAME PIC X(30) VALUE SPACES.              00002240
*                                                                  00002250
     05  FILLER              PIC X(10)  VALUE SPACES.              00002260
     05  FILLER              PIC X(18)                             00002270
             VALUE 'CARD SERIAL #: '.                             00002280
     05  DL2-SERIAL-NUMBER   PIC X(15)  VALUE SPACES.              00002290
*                                                                  00002291
 01  DETAIL-LINE3.                                                00002292
     05  FILLER              PIC X(11)  VALUE SPACES.              00002293
     05  DL3-DETAIL-DESC     PIC X(30)  VALUE SPACES.              00002294
     05  FILLER              PIC X(02)  VALUE SPACES.              00002295
     05  DL3-DETAIL-DATE     PIC X(10)  VALUE SPACES.              00002296
     05  FILLER              PIC X(04)  VALUE SPACES.              00002297
     05  DL3-DETAIL-VALUE    PIC -ZZ,ZZZ.99 VALUE ZEROES.          00002298
*                                                                  00002299
 01  TOTAL-BENES.                                                 00002300
     05  FILLER              PIC X(05)  VALUE SPACES.              00002310
     05  FILLER              PIC X(16)  VALUE 'TOTAL AUTHORIZED'.  00002320
     05  FILLER              PIC X(49)  VALUE SPACES.              00002330
     05  TB-TOTAL-AUTH       PIC -ZZZ,ZZZ.99  VALUE ZEROES.        00002340
*                                                                  00002350
 01  TOTAL-CLAIMED.                                               00002360
     05  FILLER              PIC X(05)  VALUE SPACES.              00002370
     05  FILLER              PIC X(16)  VALUE 'TOTAL CLAIMED   '.  00002380
     05  FILLER              PIC X(63)  VALUE SPACES.              00002390
     05  TC-TOTAL-CLAIMED    PIC -ZZZ,ZZZ.99  VALUE ZEROES.        00002391
*                                                                  00002392
 01  TOTAL-UNCLAIMED.                                             00002393
     05  FILLER              PIC X(05)  VALUE SPACES.              00002394
     05  FILLER              PIC X(16)  VALUE 'TOTAL UNCLAIMED '.  00002395
     05  FILLER              PIC X(77)  VALUE SPACES.              00002396
     05  TU-TOTAL-UNCLAIMED PIC -ZZZ,ZZZ.99  VALUE ZEROES.        00002397
*                                                                  00002398
 01  TOTAL-CUSTOMER.                                              00002399
     05  FILLER              PIC X(03)  VALUE SPACES.              00002400
     05  FILLER              PIC X(24)                             00002410
            VALUE 'TOTALS FOR CUSTOMER ID: '.                     00002420
     05  TCUST-CUSTOMER-ID   PIC X(14)  VALUE SPACES.              00002430
     05  FILLER              PIC X(27)  VALUE SPACES.              00002440
     05  TCUST-AUTH          PIC -Z,ZZZ,ZZZ.99 VALUE ZEROES.       00002450
     05  FILLER              PIC X      VALUE SPACES.              00002460
     05  TCUST-CLAIMED       PIC -Z,ZZZ,ZZZ.99 VALUE ZEROES.       00002470
     05  FILLER              PIC X      VALUE SPACES.              00002480
```

```
     05  TCUST-UNCLAIMED    PIC -Z,ZZZ,ZZZ.99 VALUE ZEROES.
**                                                                00002491
************************************************************      00002492
**    WORKING STORAGE VARIABLES  & ACCUMULATORS     **            00002493
************************************************************      00002494
**                                                                00002495
 01  WORKING-VARIABLES.                                           00002496
     05  WS-ABEND-PARM             PIC S9(04) COMP VALUE 0.       00002497
     05  WS-ABEND-PGM              PIC X(08)  VALUE SPACES.       00002498
                                                                  00002499
     05  WS-PRIME-KEY.                                            00002500
         10  WS-TABLE-ID     PIC 9(02)  COMP VALUE 0026.          00002510
         10  WS-PGM          PIC X(08)  VALUE 'CWRB7100'.         00002520
         10  FILLER          PIC X(07)  VALUE SPACES.             00002530
                                                                  00002540
     05  WS-CURRENT-DATE-NUM.                                     00002550
         10  CDN-CCYY               PIC 9(04) VALUE ZEROES.      00002560
         10  CDN-MM                 PIC 99    VALUE ZEROES.      00002570
         10  CDN-DD                 PIC 99    VALUE ZEROES.      00002580
         10  FILLER                 PIC X(06) VALUE SPACES.      00002590
     05  WS-DATE-HOLD                          VALUE SPACES.      00002591
         10  FILLER                 PIC XX.                      00002592
         10  DH-YY                  PIC XX.                      00002593
         10  DH-MM                  PIC XX.                      00002594
         10  DH-DD                  PIC XX.                      00002595
         10  DH-DD-NUM REDEFINES DH-DD                           00002596
                                    PIC 99.                      00002597
             88  VALID-DAY-RANGE   VALUE 01 THRU 31.             00002598
                                                                  00002599
     05  WS-MONTH                  PIC X(03)  VALUE SPACES.      00002600
         88  MJAN                             VALUE 'JAN'.       00002610
         88  MFEB                             VALUE 'FEB'.       00002620
         88  MMAR                             VALUE 'MAR'.       00002630
         88  MAPR                             VALUE 'APR'.       00002640
         88  MMAY                             VALUE 'MAY'.       00002650
         88  MJUN                             VALUE 'JUN'.       00002660
         88  MJUL                             VALUE 'JUL'.       00002670
         88  MAUG                             VALUE 'AUG'.       00002680
         88  MSEP                             VALUE 'SEP'.       00002690
         88  MOCT                             VALUE 'OCT'.       00002691
         88  MNOV                             VALUE 'NOV'.       00002692
         88  MDEC                             VALUE 'DEC'.       00002693
*                                                                 00002694
 01  ACCUMULATORS      VALUE ZEROES.                              00002695
     05  AC-TOTAL-DETAIL      PIC S9(5)V99.                       00002696
     05  AC-TOTAL-AUTHORIZED  PIC S9(6)V99.                       00002697
     05  AC-TOTAL-CLAIMED     PIC S9(6)V99.                       00002698
     05  AC-TOTAL-UNCLAIMED   PIC S9(6)V99.                       00002699
*                                                                 00002700
     05  AC-FINAL-AUTHORIZED  PIC S9(7)V99.                       00002710
     05  AC-FINAL-CLAIMED     PIC S9(7)V99.                       00002720
     05  AC-FINAL-UNCLAIMED   PIC S9(7)V99.                       00002730
     05  AC-PAGE-COUNT        PIC 9(3).                           00002740
     05  AC-LINE-COUNT        PIC 9(2).                           00002750
*                                                                 00002760
 01  WS-SWITCHES                          VALUE 'N'.             00002770
     05  WS-CUSTOMER-SW                   PIC X.                 00002780
```

```
         88  NO-MORE-CUSTOMERS              VALUE 'X'.          00002790
         88  PROCESS-CUSTOMERS              VALUE 'P'.          00002791
         88  CUSTOMER-BREAK                 VALUE 'B'.          00002792
     05  WS-CLAIMS-SW                       PIC X.              00002793
         88  EOF-CLAIMS                     VALUE 'Y'.          00002794
     05  WS-FETCH-SW                        PIC X.              00002795
         88  FIRST-FETCH                    VALUE 'F'.          00002796
         88  SUBSEQUENT-FETCH               VALUE 'S'.          00002797
/                                                               00002798
 PROCEDURE DIVISION.                                            00002799
 A000-MAIN-PROCEDURE.                                           00002800
     PERFORM B000-INITIALIZE                                    00002810
        THRU B000-EXIT.                                         00002820
                                                                00002830
     PERFORM C000-PROCESS-BENEFITS                              00002840
        THRU C000-EXIT                                          00002850
         UNTIL NO-MORE-CUSTOMERS.                               00002860
                                                                00002870
      PERFORM P700-PRINT-TOTAL-CUSTOMER                         00002880
         THRU P700-EXIT.                                        00002890
                                                                00002891
     PERFORM E000-CLEAN-UP                                      00002892
        THRU E000-EXIT.                                         00002893
 A000-EXIT.                                                     00002894
     GOBACK.                                                    00002895
                                                                00002896
 B000-INITIALIZE.                                               00002897
                                                                00002898
     OPEN INPUT REFER-FILE.                                     00002899
                                                                00002900
     IF VALID-STATUS                                            00002910
        PERFORM V100-READ-REFER-FILE                            00002920
           THRU V100-EXIT                                       00002930
     ELSE                                                       00002940
        DISPLAY '*** ERROR OPENING REFERENCE FILE ***'          00002950
          ' STATUS: ' RF-STATUS                                 00002960
        GO TO Z9999-ABEND                                       00002970
     END-IF.                                                    00002980
                                                                00002990
     OPEN OUTPUT RPT-FILE.                                      00002991
                                                                00002992
     MOVE WS-PGM             TO HL1-PROGRAM-NME.                00002993
                                                                00002994
                                                                00002995
     INITIALIZE ACCUMULATORS.                                   00002996
                                                                00002997
     EXEC SQL                                                   00002998
       SELECT TO_CHAR(SYSDATE, 'YYYYMMDDHHMMSS')                00002999
         INTO :WS-CURRENT-DATE                                  00003000
         FROM DUAL                                              00003010
     END-EXEC.                                                  00003020
                                                                00003030
                                                                00003040
     MOVE WS-CURRENT-DATE  TO WS-CURRENT-DATE-NUM.              00003050
     STRING WS-CURRENT-DATE(9:2), ':',                          00003060
            WS-CURRENT-DATE(11:2), ':',                         00003070
            WS-CURRENT-DATE(13:2)                               00003080
```

```
                DELIMITED BY SIZE                                       00003090
                   INTO HL1-RUN-TIME                                    00003091
                                                                        00003092
    MOVE CDN-CCYY       TO HL1-RD-YEAR.                                 00003093
    MOVE CDN-MM         TO HL1-RD-MONTH.                                00003094
    MOVE CDN-DD         TO HL1-RD-DAY.                                  00003095
                                                                        00003096
    PERFORM V110-OPEN-BENEFITS                                          00003097
       THRU V110-EXIT.                                                  00003098
                                                                        00003099
    IF PROCESS-CUSTOMERS AND ORA-SQL-SUCCESSFUL                         00003100
        PERFORM P100-PRINT-HEADER1                                      00003110
           THRU P100-EXIT                                               00003120
        PERFORM V115-FETCH-BENEFIT                                      00003130
           THRU V115-EXIT                                               00003140
        IF NOT (NO-MORE-CUSTOMERS AND FIRST-FETCH)                      00003150
            PERFORM P200-PRINT-CUST-INFO                                00003160
               THRU P200-EXIT                                           00003170
            PERFORM P300-PRINT-CARDHLDR-INFO                            00003180
               THRU P300-EXIT                                           00003190
            SET SUBSEQUENT-FETCH TO TRUE                                00003191
        END-IF                                                          00003192
    END-IF.                                                             00003193
B000-EXIT.                                                              00003194
     EXIT.                                                              00003195
                                                                        00003196
                                                                        00003197
C000-PROCESS-BENEFITS.                                                  00003198
    IF NOT (BEN-CUSTOMER-ID = WS-OLD-CUSTOMER-ID)                       00003199
      PERFORM C100-PROCESS-CUSTOMER-BREAK                               00003200
         THRU C100-EXIT                                                 00003210
    END-IF.                                                             00003220
                                                                        00003230
    IF NOT (BEN-SERIAL-NUM = WS-OLD-SERIAL-NUMBER)                      00003240
       PERFORM C200-PROCESS-CARDHOLDER-BREAK                            00003250
          THRU C200-EXIT                                                00003260
    END-IF.                                                             00003270
                                                                        00003280
    MOVE BEN-EFFECTIVE-DATE   TO DL3-DETAIL-DATE.                       00003290
    MOVE BEN-INITIAL-VAL-AMT  TO AC-TOTAL-DETAIL.                       00003291
    MOVE AC-TOTAL-DETAIL      TO DL3-DETAIL-VALUE.                      00003292
    MOVE BEN-BENEFIT-TYPE     TO DEF-BENEFIT-TYPE.                      00003293
                                                                        00003294
    PERFORM C350-OBTAIN-DEFINITION                                      00003295
       THRU C350-EXIT                                                   00003296
                                                                        00003297
    MOVE DEF-BENEFIT-DESC     TO DL3-DETAIL-DESC.                       00003298
    ADD  BEN-INITIAL-VAL-AMT  TO AC-TOTAL-AUTHORIZED                    00003299
                                 AC-FINAL-AUTHORIZED.                   00003300
                                                                        00003310
    PERFORM P400-PRINT-DETAIL                                           00003320
       THRU P400-EXIT                                                   00003330
                                                                        00003340
    PERFORM V115-FETCH-BENEFIT                                          00003350
       THRU V115-EXIT.                                                  00003360
                                                                        00003370
C000-EXIT.                                                              00003380
```

```
     EXIT.                                                          00003390
                                                                    00003391
C100-PROCESS-CUSTOMER-BREAK.                                        00003392
*    DISPLAY 'C100-PROCESS-CUSTOMER-BREAK'                          00003393
     SET CUSTOMER-BREAK TO TRUE                                     00003394
     PERFORM C200-PROCESS-CARDHOLDER-BREAK                          00003395
        THRU C200-EXIT                                              00003396
     PERFORM P700-PRINT-TOTAL-CUSTOMER                              00003397
        THRU P700-EXIT                                              00003398
     PERFORM P100-PRINT-HEADER1                                     00003399
        THRU P100-EXIT                                              00003400
                                                                    00003410
     MOVE BEN-SERIAL-NUM      TO WS-OLD-SERIAL-NUMBER.              00003420
     MOVE BEN-CUSTOMER-ID     TO WS-OLD-CUSTOMER-ID.                00003430
                                                                    00003440
     PERFORM P200-PRINT-CUST-INFO                                   00003450
        THRU P200-EXIT.                                             00003460
     PERFORM P300-PRINT-CARDHLDR-INFO                               00003470
        THRU P300-EXIT.                                             00003480
     MOVE SPACES TO WS-CUSTOMER-SW.                                 00003490
C100-EXIT.                                                          00003491
     EXIT.                                                          00003492
                                                                    00003493
C200-PROCESS-CARDHOLDER-BREAK.                                      00003494
     PERFORM P410-PRINT-TOTAL-BENES                                 00003495
        THRU P410-EXIT.                                             00003496
     PERFORM V210-OPEN-CLAIMS                                       00003497
        THRU V210-EXIT.                                             00003498
                                                                    00003499
     PERFORM UNTIL EOF-CLAIMS                                       00003500
         MOVE CLM-REQUEST-DT-TM     TO DL3-DETAIL-DATE              00003510
         MOVE CLM-CLAIM-VAL-AMT    TO DL3-DETAIL-VALUE              00003520
         MOVE CLM-BENEFIT-TYPE      TO DEF-BENEFIT-TYPE             00003530
                                                                    00003540
         PERFORM C350-OBTAIN-DEFINITION                             00003550
            THRU C350-EXIT                                          00003560
                                                                    00003570
         MOVE DEF-BENEFIT-DESC      TO DL3-DETAIL-DESC              00003580
         ADD CLM-CLAIM-VAL-AMT      TO AC-TOTAL-CLAIMED,            00003590
                                       AC-FINAL-CLAIMED             00003591
          PERFORM P400-PRINT-DETAIL                                 00003592
             THRU P400-EXIT                                         00003593
                                                                    00003594
          PERFORM V220-FETCH-CLAIMS                                 00003595
             THRU V220-EXIT                                         00003596
     END-PERFORM.                                                   00003597
                                                                    00003598
     PERFORM V260-CLOSE-CLAIMS                                      00003599
        THRU V260-EXIT.                                             00003600
                                                                    00003610
     PERFORM P500-PRINT-TOTAL-CLAIMS                                00003620
        THRU P500-EXIT.                                             00003630
                                                                    00003640
     PERFORM P600-PRINT-TOTAL-UNCLAIMED                             00003650
        THRU P600-EXIT.                                             00003660
                                                                    00003670
     IF NOT (CUSTOMER-BREAK OR NO-MORE-CUSTOMERS)                   00003680
```

```
        PERFORM P300-PRINT-CARDHLDR-INFO                          00003690
           THRU P300-EXIT                                         00003691
    END-IF.                                                       00003692
    MOVE BEN-SERIAL-NUM      TO WS-OLD-SERIAL-NUMBER.             00003693
C200-EXIT.                                                        00003694
     EXIT.                                                        00003695
                                                                  00003696
C350-OBTAIN-DEFINITION.                                           00003697
    INITIALIZE DL3-DETAIL-DESC,                                   00003698
               DEF-BENEFIT-DESC.                                  00003699
                                                                  00003700
    EXEC SQL                                                      00003710
        SELECT BENEFIT_DESC                                       00003720
          INTO :DEF-BENEFIT-DESC                                  00003730
          FROM MCHECK.BENEFITS_DEFINITION                         00003740
         WHERE BENEFIT_TYPE = :DEF-BENEFIT-TYPE                   00003750
    END-EXEC.                                                     00003760
                                                                  00003770
    MOVE SQLCODE   TO ORA-NAMED-SQLCODE                           00003780
                                                                  00003790
    EVALUATE TRUE                                                 00003791
       WHEN ORA-SQL-SUCCESSFUL                                    00003792
                   MOVE DEF-BENEFIT-DESC TO DL3-DETAIL-DESC       00003793
                                                                  00003794
       WHEN ORA-SQL-ROW-NOT-FOUND                                 00003795
                   INITIALIZE DEF-BENEFIT-DESC                    00003796
                   MOVE ' NO DESC AVAIL '                         00003797
                                TO DEF-BENEFIT-DESC               00003798
       WHEN OTHER                                                 00003799
                            DISPLAY                               00003800
                  '* INVALID SELECT ON DEFINITIONS FILE '         00003810
                            GO TO Z9999-ABEND                     00003820
    END-EVALUATE.                                                 00003830
C350-EXIT.                                                        00003840
     EXIT.                                                        00003850
                                                                  00003860
E000-CLEAN-UP.                                                    00003870
                                                                  00003880
    INITIALIZE RPT-REC                                            00003890
    WRITE RPT-REC.                                                00003891
                                                                  00003892
    DISPLAY '**  EOJ ' WS-PGM                                     00003893
    DISPLAY '    '                                                00003894
    DISPLAY '**'                                                  00003895
                                                                  00003896
    EXEC SQL                                                      00003897
       CLOSE BENE                                                 00003898
    END-EXEC.                                                     00003899
    EXEC SQL                                                      00003900
       CLOSE CLMS                                                 00003910
    END-EXEC.                                                     00003920
                                                                  00003930
    CLOSE RPT-FILE, REFER-FILE.                                   00003940
E000-EXIT.                                                        00003950
     EXIT.                                                        00003960
                                                                  00003970
P100-PRINT-HEADER1.                                               00003980
```

```
     ADD 1                TO AC-PAGE-COUNT.                          00003990
     MOVE AC-PAGE-COUNT   TO HL1-PAGE-NUMBER.                        00003991
     MOVE WS-START-EXPIRATION-DATE TO HL1-START-DATE                 00003992
     MOVE WS-END-EXPIRATION-DATE   TO HL1-END-DATE.                  00003993
                                                                     00003994
     INITIALIZE RPT-REC, AC-LINE-COUNT                               00003995
     WRITE RPT-REC AFTER ADVANCING TOP-OF-PAGE.                      00003996
     WRITE RPT-REC FROM HEADING-LINE1                                00003997
     WRITE RPT-REC FROM HEADING-LINE1A                               00003998
     WRITE RPT-REC FROM HEADING-LINE1B                               00003999
     INITIALIZE RPT-REC.                                             00004000
     WRITE RPT-REC.                                                  00004010
     ADD 5 TO AC-LINE-COUNT.                                         00004020
P100-EXIT.                                                           00004030
     EXIT.                                                           00004040
                                                                     00004050
P200-PRINT-CUST-INFO.                                                00004060
    PERFORM P900-CHECK-PAGE-BREAK.                                   00004070
    INITIALIZE HEADING-LINE2, HEADING-LINE3, CUSTOMER-ROW,           00004080
               HEADING-LINE4, HEADING-LINE5.                         00004090
                                                                     00004091
    MOVE BEN-CUSTOMER-ID    TO HL2-CUSTOMER-ID                       00004092
                               WS-OLD-CUSTOMER-ID.                   00004093
                                                                     00004094
    EXEC SQL                                                         00004095
      SELECT CUSTOMER_NAME,                                          00004096
             STREET_ADDR1,                                           00004097
             STREET_ADDR2,                                           00004098
             CITY,                                                   00004099
             STATE,                                                  00004100
             ZIPCODE                                                 00004110
        INTO :CUST-CUSTOMER-NAME,                                    00004120
             :CUST-STREET-ADDR1,                                     00004130
             :CUST-STREET-ADDR2,                                     00004140
             :CUST-CITY,                                             00004150
             :CUST-STATE,                                            00004160
             :CUST-ZIPCODE                                           00004170
      FROM MCHECK.CUSTOMER                                           00004180
      WHERE CUSTOMER_ID = :BEN-CUSTOMER-ID                           00004190
    END-EXEC.                                                        00004191
                                                                     00004192
    MOVE SQLCODE   TO ORA-NAMED-SQLCODE                              00004193
                                                                     00004194
    EVALUATE TRUE                                                    00004195
       WHEN ORA-SQL-SUCCESSFUL                                       00004196
                            CONTINUE                                 00004197
                                                                     00004198
       WHEN ORA-SQL-ROW-NOT-FOUND                                    00004199
                            INITIALIZE CUSTOMER-ROW                  00004200
                            MOVE ' NO CUST INFO AVAIL '              00004210
                              TO CUST-CUSTOMER-NAME                  00004220
       WHEN OTHER                                                    00004230
                            DISPLAY                                  00004240
                  '* INVALID SELECT ON CUSTOMER FILE '               00004250
                            GO TO Z9999-ABEND                        00004260
    END-EVALUATE.                                                    00004270
                                                                     00004280
```

```
    MOVE CUST-CUSTOMER-NAME   TO HL2-CUSTOMER-NAME.                00004290
    INITIALIZE RPT-REC.                                           00004291
    WRITE RPT-REC.                                                00004292
    WRITE RPT-REC FROM HEADING-LINE2                              00004293
    ADD 2 TO AC-LINE-COUNT.                                       00004294
                                                                  00004295
    IF ORA-SQL-SUCCESSFUL                                         00004296
      INITIALIZE RPT-REC, HL3-CUST-ADDR1                          00004297
      MOVE CUST-STREET-ADDR1 TO HL3-CUST-ADDR1                    00004298
      WRITE RPT-REC FROM HEADING-LINE3                            00004299
      ADD 1 TO AC-LINE-COUNT                                      00004300
                                                                  00004310
      IF CUST-STREET-ADDR2 > SPACES                               00004320
         INITIALIZE RPT-REC, HL4-CUST-ADDR2                       00004330
         MOVE CUST-STREET-ADDR2 TO HL4-CUST-ADDR2                 00004340
         WRITE RPT-REC FROM HEADING-LINE4                         00004350
         ADD 1 TO AC-LINE-COUNT                                   00004360
      END-IF                                                      00004370
                                                                  00004380
      INITIALIZE  RPT-REC, HL5-CITY-STATE-ZIP                     00004390
      STRING CUST-CITY ', ' CUST-STATE '  ' CUST-ZIPCODE          00004391
            DELIMITED BY SIZE                                     00004392
         INTO HL5-CITY-STATE-ZIP                                  00004393
      WRITE RPT-REC FROM HEADING-LINE5                            00004394
      ADD 1 TO AC-LINE-COUNT                                      00004395
    END-IF                                                        00004396
                                                                  00004397
    INITIALIZE RPT-REC                                            00004398
    WRITE RPT-REC.                                                00004399
    WRITE RPT-REC FROM DETAIL-HEADING                             00004400
    ADD 2 TO AC-LINE-COUNT.                                       00004410
P200-EXIT.                                                        00004420
     EXIT.                                                        00004430
                                                                  00004440
P300-PRINT-CARDHLDR-INFO.                                         00004450
     PERFORM P900-CHECK-PAGE-BREAK.                               00004460
     MOVE BEN-SERIAL-NUM     TO WS-OLD-SERIAL-NUMBER,             00004470
                                DL2-SERIAL-NUMBER.                00004480
     MOVE BEN-MFG-SERIAL-NUM TO WS-OLD-MFG-SERIAL-NUMBER.         00004490
                                                                  00004491
     INITIALIZE DL1-CARDHOLDER-NAME.                              00004492
                                                                  00004493
     EXEC SQL                                                     00004494
        SELECT LAST_NAME,                                         00004495
               FIRST_NAME                                         00004496
         INTO :CRDH-LAST-NAME,                                    00004497
              :CRDH-FIRST-NAME                                    00004498
         FROM MCHECK.CARDHOLDER                                   00004499
        WHERE SERIAL_NUM = :BEN-SERIAL-NUM                        00004500
     END-EXEC.                                                    00004510
                                                                  00004520
    MOVE SQLCODE   TO ORA-NAMED-SQLCODE.                          00004530
                                                                  00004540
    EVALUATE TRUE                                                 00004550
       WHEN ORA-SQL-SUCCESSFUL                                    00004560
            STRING CRDH-LAST-NAME, ', ',  CRDH-FIRST-NAME         00004570
              DELIMITED BY SIZE                                   00004580
```

```
                INTO DL1-CARDHOLDER-NAME                                00004590
                                                                        00004591
           WHEN ORA-SQL-ROW-NOT-FOUND                                   00004592
                      MOVE '** NO CARDHOLDER NAME AVAIL **'             00004593
                         TO DL1-CARDHOLDER-NAME                         00004594
           WHEN OTHER                                                   00004595
                   DISPLAY                                              00004596
                     '* INVALID SELECT ON CARDHOLDER FILE '             00004597
                           '   SQLCODE: ' ORA-NAMED-SQLCODE             00004598
                    GO TO Z9999-ABEND                                   00004599
        END-EVALUATE.                                                   00004600
                                                                        00004610
        INITIALIZE RPT-REC.                                             00004620
        WRITE RPT-REC                                                   00004630
        WRITE RPT-REC FROM DETAIL-LINE1                                 00004640
        ADD 2 TO AC-LINE-COUNT.                                         00004650
P300-EXIT.                                                              00004660
        EXIT.                                                           00004670
                                                                        00004680
P400-PRINT-DETAIL.                                                      00004690
    PERFORM P900-CHECK-PAGE-BREAK.                                      00004691
    INITIALIZE RPT-REC                                                  00004692
    WRITE RPT-REC FROM DETAIL-LINE3.                                    00004693
    ADD 1 TO AC-LINE-COUNT.                                             00004694
P400-EXIT.                                                              00004695
    EXIT.                                                               00004696
                                                                        00004697
P410-PRINT-TOTAL-BENES.                                                 00004698
    PERFORM P900-CHECK-PAGE-BREAK.                                      00004699
    MOVE AC-TOTAL-AUTHORIZED      TO TB-TOTAL-AUTH.                     00004700
    INITIALIZE RPT-REC                                                  00004710
    WRITE RPT-REC FROM TOTAL-BENES.                                     00004720
    ADD 1 TO AC-LINE-COUNT.                                             00004730
P410-EXIT.                                                              00004740
     EXIT.                                                              00004750
                                                                        00004760
P500-PRINT-TOTAL-CLAIMS.                                                00004770
    PERFORM P900-CHECK-PAGE-BREAK.                                      00004780
    MOVE AC-TOTAL-CLAIMED         TO TC-TOTAL-CLAIMED.                  00004790
    INITIALIZE RPT-REC                                                  00004791
    WRITE RPT-REC FROM TOTAL-CLAIMED.                                   00004792
    ADD 1 TO AC-LINE-COUNT.                                             00004793
P500-EXIT.                                                              00004794
     EXIT.                                                              00004795
                                                                        00004796
P600-PRINT-TOTAL-UNCLAIMED.                                             00004797
    PERFORM P900-CHECK-PAGE-BREAK.                                      00004798
    SUBTRACT AC-TOTAL-CLAIMED FROM AC-TOTAL-AUTHORIZED                  00004799
            GIVING AC-TOTAL-UNCLAIMED                                   00004800
    ADD AC-TOTAL-UNCLAIMED TO AC-FINAL-UNCLAIMED.                       00004810
                                                                        00004820
    MOVE AC-TOTAL-UNCLAIMED       TO TU-TOTAL-UNCLAIMED.                00004830
    INITIALIZE RPT-REC                                                  00004840
    WRITE RPT-REC FROM TOTAL-UNCLAIMED.                                 00004850
    ADD 1 TO AC-LINE-COUNT.                                             00004860
                                                                        00004870
    INITIALIZE AC-TOTAL-AUTHORIZED,                                     00004880
```

```
                AC-TOTAL-CLAIMED,                                       00004890
                AC-TOTAL-UNCLAIMED.                                     00004891
P600-EXIT.                                                              00004892
     EXIT.                                                              00004893
                                                                        00004894
P700-PRINT-TOTAL-CUSTOMER.                                              00004895
    PERFORM P900-CHECK-PAGE-BREAK.                                      00004896
    MOVE AC-FINAL-AUTHORIZED   TO TCUST-AUTH                            00004897
    MOVE AC-FINAL-CLAIMED      TO TCUST-CLAIMED                         00004898
    MOVE AC-FINAL-UNCLAIMED    TO TCUST-UNCLAIMED                       00004899
    MOVE HL2-CUSTOMER-ID       TO TCUST-CUSTOMER-ID                     00004900
                                                                        00004910
    INITIALIZE RPT-REC                                                  00004920
    WRITE RPT-REC                                                       00004930
    WRITE RPT-REC FROM TOTAL-CUSTOMER                                   00004940
    ADD 2 TO AC-LINE-COUNT                                              00004950
                                                                        00004960
    INITIALIZE AC-FINAL-AUTHORIZED,                                     00004970
                AC-FINAL-CLAIMED,                                       00004980
                AC-FINAL-UNCLAIMED.                                     00004990
P700-EXIT.                                                              00004991
     EXIT.                                                              00004992
                                                                        00004993
P900-CHECK-PAGE-BREAK.                                                  00004994
    IF AC-LINE-COUNT > 61                                               00004995
        PERFORM P100-PRINT-HEADER1                                      00004996
           THRU P100-EXIT                                               00004997
        INITIALIZE RPT-REC                                              00004998
        WRITE RPT-REC                                                   00004999
        WRITE RPT-REC FROM DETAIL-HEADING                               00005000
        INITIALIZE RPT-REC                                              00005010
        WRITE RPT-REC                                                   00005020
        ADD 3 TO AC-LINE-COUNT                                          00005030
    END-IF.                                                             00005040
P900-EXIT.                                                              00005050
                                                                        00005060
V100-READ-REFER-FILE.                                                   00005070
                                                                        00005080
    MOVE WS-PRIME-KEY TO RF-PRIME-KEY                                   00005090
                                                                        00005091
    READ REFER-FILE                                                     00005092
      KEY    IS RF-PRIME-KEY.                                           00005093
                                                                        00005094
    IF VALID-STATUS                                                     00005095
      MOVE RF-START-EXPIRATION-DATE   TO WS-DATE-HOLD                   00005096
      PERFORM V105-EDIT-INPUT-DATES                                     00005097
         THRU V105-EXIT                                                 00005098
      STRING DH-DD '-' WS-MONTH '-' DH-YY                               00005099
            DELIMITED BY SIZE                                           00005100
          INTO WS-START-EXPIRATION-DATE                                 00005110
                                                                        00005120
      MOVE RF-END-EXPIRATION-DATE   TO WS-DATE-HOLD                     00005130
      PERFORM V105-EDIT-INPUT-DATES                                     00005140
         THRU V105-EXIT                                                 00005150
      STRING DH-DD '-' WS-MONTH '-' DH-YY                               00005160
            DELIMITED BY SIZE                                           00005170
          INTO WS-END-EXPIRATION-DATE                                   00005180
```

```
    ELSE                                                          00005190
      DISPLAY '*** ERROR READING REFERENCE FILE ***'              00005191
       ' STATUS: ' RF-STATUS                                      00005192
       '  RF-PRIME-KEY: ' RF-PRIME-KEY                            00005193
       '  WS-PRIME-KEY: ' WS-PRIME-KEY                            00005194
      GO TO Z9999-ABEND                                           00005195
    END-IF.                                                       00005196
V100-EXIT.                                                        00005197
     EXIT.                                                        00005198
                                                                  00005199
                                                                  00005200
V105-EDIT-INPUT-DATES.                                            00005210
    IF DH-DD IS NUMERIC                                           00005220
       IF NOT VALID-DAY-RANGE                                     00005230
         DISPLAY                                                  00005240
           '***  ERROR PROCESSING INPUT DATE PARMS ***'           00005250
            ' INVALID DAY SPECIFIED '                             00005260
                        GO TO Z9999-ABEND                         00005270
       END-IF                                                     00005280
    END-IF.                                                       00005290
    IF DH-YY IS NOT NUMERIC                                       00005291
        DISPLAY                                                   00005292
           '***  ERROR PROCESSING INPUT DATE PARMS ***'           00005293
            ' INVALID YEAR SPECIFIED '                            00005294
                        GO TO Z9999-ABEND                         00005295
    END-IF.                                                       00005296
    EVALUATE DH-MM                                                00005297
       WHEN '01'            SET MJAN TO TRUE                      00005298
       WHEN '02'            SET MFEB TO TRUE                      00005299
       WHEN '03'            SET MMAR TO TRUE                      00005300
       WHEN '04'            SET MAPR TO TRUE                      00005310
       WHEN '05'            SET MMAY TO TRUE                      00005320
       WHEN '06'            SET MJUN TO TRUE                      00005330
       WHEN '07'            SET MJUL TO TRUE                      00005340
       WHEN '08'            SET MAUG TO TRUE                      00005350
       WHEN '09'            SET MSEP TO TRUE                      00005360
       WHEN '10'            SET MOCT TO TRUE                      00005370
       WHEN '11'            SET MNOV TO TRUE                      00005380
       WHEN '12'            SET MDEC TO TRUE                      00005390
       WHEN OTHER           DISPLAY                               00005391
           '***  ERROR PROCESSING INPUT DATE PARMS ***'           00005392
            ' INVALID MONTH SPECIFIED '                           00005393
                        GO TO Z9999-ABEND                         00005394
    END-EVALUATE.                                                 00005395
V105-EXIT.                                                        00005396
     EXIT.                                                        00005397
                                                                  00005398
V110-OPEN-BENEFITS.                                               00005399
    EXEC SQL                                                      00005400
      OPEN BENE                                                   00005410
    END-EXEC.                                                     00005420
                                                                  00005430
    MOVE SQLCODE   TO ORA-NAMED-SQLCODE                           00005440
    EVALUATE TRUE                                                 00005450
       WHEN ORA-SQL-SUCCESSFUL                                    00005460
                            SET PROCESS-CUSTOMERS TO TRUE         00005470
                            SET FIRST-FETCH TO TRUE               00005480
```

```
                                                                        00005490
                                                                        00005491
       WHEN OTHER          DISPLAY 'BAD OPEN ON BENEFITS FILE'          00005492
                                    GO TO Z9999-ABEND                   00005493
    END-EVALUATE.                                                       00005494
V110-EXIT.                                                              00005495
     EXIT.                                                              00005496
                                                                        00005497
V115-FETCH-BENEFIT.                                                     00005498
    EXEC SQL                                                            00005499
       FETCH BENE                                                       00005500
        INTO :BEN-EFFECTIVE-DATE,                                       00005510
             :BEN-SERIAL-NUM,                                           00005520
             :BEN-CUSTOMER-ID,                                          00005530
             :BEN-BENEFIT-TYPE,                                         00005540
             :BEN-EXPIRATION-DATE,                                      00005550
             :BEN-MFG-SERIAL-NUM,                                       00005560
             :BEN-INITIAL-VAL-AMT                                       00005570
    END-EXEC.                                                           00005580
*                                                                       00005590
    MOVE SQLCODE   TO ORA-NAMED-SQLCODE                                 00005591
                                                                        00005592
    EVALUATE TRUE                                                       00005593
       WHEN ORA-SQL-SUCCESSFUL                                          00005594
                            CONTINUE                                    00005595
                                                                        00005596
       WHEN ORA-SQL-ROW-NOT-FOUND                                       00005597
         SET NO-MORE-CUSTOMERS TO TRUE                                  00005598
         IF FIRST-FETCH                                                 00005599
          DISPLAY                                                       00005600
         'NO CUSTOMER BENEFITS FOUND FOR PERIODS SPECIFIED'             00005610
          DISPLAY '   START DATE: ' WS-START-EXPIRATION-DATE            00005620
          DISPLAY '   END DATE:   ' WS-END-EXPIRATION-DATE             00005630
                                                                        00005640
          INITIALIZE RPT-REC                                            00005650
          MOVE                                                          00005660
       '*************************************************************' 00005670
              TO RPT-REC                                                00005680
          WRITE RPT-REC                                                 00005690
                                                                        00005691
          INITIALIZE RPT-REC                                            00005692
          MOVE                                                          00005693
       '**                                                         **'  00005694
              TO RPT-REC                                                00005695
          WRITE RPT-REC                                                 00005696
                                                                        00005697
          INITIALIZE RPT-REC                                            00005698
          MOVE                                                          00005699
       '** NO CUSTOMER BENEFITS FOUND FOR PERIODS SPECIFIED  **'        00005700
              TO RPT-REC                                                00005710
          WRITE RPT-REC                                                 00005720
                                                                        00005730
          INITIALIZE RPT-REC                                            00005740
         STRING '**     START DATE: ' WS-START-EXPIRATION-DATE          00005750
                '                          **'                         00005760
           DELIMITED BY SIZE                                            00005770
            INTO RPT-REC                                                00005780
```

```
           WRITE RPT-REC                                                00005790
                                                                        00005791
           INITIALIZE RPT-REC                                           00005792
           STRING '**     END DATE:    ' WS-END-EXPIRATION-DATE        00005793
                  '                          **'                       00005794
             DELIMITED BY SIZE                                          00005795
              INTO RPT-REC                                              00005796
           WRITE RPT-REC                                                00005797
                                                                        00005798
           INITIALIZE RPT-REC                                           00005799
           MOVE                                                         00005800
       '**                                                      **'      00005810
               TO RPT-REC                                               00005820
           WRITE RPT-REC                                                00005830
                                                                        00005840
           INITIALIZE RPT-REC                                           00005850
           MOVE                                                         00005860
       '************************************************************'   00005870
                  TO RPT-REC                                            00005880
              WRITE RPT-REC                                             00005890
              INITIALIZE RPT-REC                                        00005891
              WRITE RPT-REC                                             00005892
           ELSE                                                         00005893
              PERFORM                                                   00005894
                C200-PROCESS-CARDHOLDER-BREAK                           00005895
                   THRU C200-EXIT                                       00005896
           END-IF                                                       00005897
                                                                        00005898
         WHEN OTHER                                                     00005899
                              DISPLAY                                   00005900
                    '* INVALID FETCH ON BENEFITS FILE '                 00005910
                              GO TO Z9999-ABEND                         00005920
       END-EVALUATE.                                                    00005930
   V115-EXIT.                                                           00005940
        EXIT.                                                           00005950
                                                                        00005960
   V210-OPEN-CLAIMS.                                                    00005970
                                                                        00005980
       EXEC SQL                                                         00005990
         OPEN CLMS                                                      00005991
       END-EXEC.                                                        00005992
                                                                        00005993
       MOVE SQLCODE   TO ORA-NAMED-SQLCODE                              00005994
                                                                        00005995
       EVALUATE TRUE                                                    00005996
          WHEN ORA-SQL-SUCCESSFUL                                       00005997
                               MOVE 'N' TO WS-CLAIMS-SW                 00005998
                               PERFORM V220-FETCH-CLAIMS                00005999
                                                                        00006000
          WHEN ORA-SQL-ROW-NOT-FOUND                                    00006010
                               SET EOF-CLAIMS  TO TRUE                  00006020
          WHEN OTHER                                                    00006030
                        DISPLAY 'INVALID OPEN ON CLAIMS FILE'           00006040
                               GO TO Z9999-ABEND                        00006050
       END-EVALUATE.                                                    00006060
   V210-EXIT.                                                           00006070
        EXIT.                                                           00006080
```

```
                                                                        00006090
V220-FETCH-CLAIMS.                                                      00006091
    EXEC SQL                                                            00006092
        FETCH CLMS                                                      00006093
         INTO :CLM-REQUEST-DT-TM,                                       00006094
              :CLM-EXPIRATION-DATE,                                     00006095
              :CLM-SERIAL-NUM,                                          00006096
              :CLM-MFG-SERIAL-NUM,                                      00006097
              :CLM-BENEFIT-TYPE,                                        00006098
              :CLM-CLAIM-VAL-AMT                                        00006099
     END-EXEC.                                                          00006100
                                                                        00006110
    MOVE SQLCODE   TO ORA-NAMED-SQLCODE                                 00006120
                                                                        00006130
    EVALUATE TRUE                                                       00006140
       WHEN ORA-SQL-SUCCESSFUL                                          00006150
                             CONTINUE                                   00006160
                                                                        00006170
       WHEN ORA-SQL-ROW-NOT-FOUND                                       00006180
                             SET EOF-CLAIMS TO TRUE                     00006190
       WHEN OTHER                                                       00006191
                             DISPLAY                                    00006192
                    '* INVALID FETCH ON CLAIMS FILE '                   00006193
                             GO TO Z9999-ABEND                          00006194
    END-EVALUATE.                                                       00006195
V220-EXIT.                                                              00006196
     EXIT.                                                              00006197
                                                                        00006198
V260-CLOSE-CLAIMS.                                                      00006199
    EXEC SQL                                                            00006200
       CLOSE CLMS                                                       00006210
    END-EXEC.                                                           00006220
                                                                        00006230
    MOVE SQLCODE TO ORA-NAMED-SQLCODE.                                  00006240
    IF ORA-SQL-SUCCESSFUL                                               00006250
         CONTINUE                                                       00006260
    ELSE                                                                00006270
        DISPLAY 'INVALID CLOSE ON CLAIMS FILE - SQLCODE: '              00006280
           ORA-NAMED-SQLCODE       SQLCODE                              00006290
        GO TO Z9999-ABEND                                               00006291
    END-IF.                                                             00006292
V260-EXIT.                                                              00006293
     EXIT.                                                              00006294
                                                                        00006295
Z9999-ABEND.                                                            00006296
                                                                        00006297
    DISPLAY '***'                                                       00006298
    DISPLAY 'ABENDING PROGRAM DUE TO PROCESSING ERRORS!!'               00006299
                                                                        00006300
    DISPLAY 'ABENDING SQLCODE: ' SQLCODE                                00006310
        '   '  ORA-NAMED-SQLCODE                                        00006320
    INITIALIZE MSG-TEXT.                                                00006330
    CALL 'SQLGLM' USING MSG-TEXT, MAX-SIZE, MSG-LENGTH.                 00006340
    DISPLAY 'ABEND SQL MSG: ' MSG-TEXT                                  00006350
    DISPLAY ' '                                                         00006360
                                                                        00006370
                                                                        00006380
```

```
                                                                        00006390
           INITIALIZE RPT-REC.                                          00006391
           STRING 'ABENDING PROGRAM '  WS-PGM                           00006392
                ' DUE TO PROCESSING ERRORS!!'                           00006393
                    DELIMITED BY SIZE                                   00006394
                         INTO RPT-REC                                   00006395
           WRITE RPT-REC.                                               00006396
                                                                        00006397
                                                                        00006398
           MOVE 999  TO WS-ABEND-PARM.                                  00006399
           DISPLY WS-ABEND-PARM.                                        00006400
                                                                        00006410
           PERFORM E000-CLEAN-UP                                        00006420
              THRU E000-EXIT.                                           00006430
                                                                        00006440
           MOVE 'ILBOABNO' TO WS-ABEND-PGM.                             00006450
           CALL WS-ABEND-PGM  USING WS-ABEND-PARM.                      00006460
       Z9999-EXIT.                                                      00006470
             EXIT.                                                      00006480
```

APPENDIX D

```
      ************************************************************************00000100
       IDENTIFICATION DIVISION.                                                00000200
      ************************************************************************00000300
       PROGRAM-ID.    CWRB7110.                                                00000400
       AUTHOR.        CUBIC/CARCG.                                             00000500
       INSTALLATION.                                                           00000600
       DATE-WRITTEN.  FEBRUARY 2000.                                           00000700
       DATE-COMPILED.                                                          00000800
      ************************************************************************00000900
      *                                                                       *00001000
      * PROGRAM NAME: BENEFITS REPORT PGM                                     *00001100
      * PROGRAM ID:   CWRB7110                                                *00001200
      *                                                                       *00001300
      * SYSTEM:  9121-490, MVS/XA,  CICS, COBOL II, VSAM                      *00001400
      * PROJECT: 170-2719, ELECTRONIC BENEFITS DISTRIBUTION SYSTEM            *00001500
      *                    BATCH REPORTS.                                     *00001600
      *                                                                       *00001700
      * DESC:    THIS PROGRAM WILL SCAN THE BENEFITS TABLE FOR ROWS           *00001800
      *          LOADED WITHIN A SPECIFIED DATE RANGE.                        *00001900
      *          THIS REPORT IS PRODUCED BY CUSTOMER ID FOR BILLING           *00002000
      *          BENEFITS TO THAT CUSTOMER.                                   *00002100
      *                                                                       *00002200
      * INPUTS:                                                               *00002300
      *          REFERENCE FILE     DEFMF03                                   *00002400
      *          BENEFITS    TABLE  (ORACLE )                                 *00002500
      *          CUSTOMER    TABLE  (ORACLE )                                 *00002600
      *                                                                       *00002700
      * OUTPUTS:                                                              *00002800
      *          A/R BENEFITS CUSTOMER BILLING REPORT                         *00002900
      *                                                                       *00003000
      * ERRORS:                                                               *00003100
      *                                                                       *00003200
      ************************************************************************00003300
      * REVISION HISTORY:                                                     *00003400
      *                                                                       *00003500
      ************************************************************************00003600
      *                                                                       *00003700
      * 12/22/99 SMB000 JXK  INITIAL CODING                                   *00003800
      *-----------------------------------------------------------------------*00003900
      ************************************************************************00004000
      /                                                                        00004100
       ENVIRONMENT DIVISION.                                                   00004200
       CONFIGURATION SECTION.                                                  00004300
       SPECIAL-NAMES.   C01 IS TOP-OF-PAGE.                                    00004400
*******SOURCE-COMPUTER. ES-9000 WITH DEBUGGING MODE.                           00004500
       SOURCE-COMPUTER. ES-9000.                                               00004600
       OBJECT-COMPUTER. ES-9000.                                               00004700
                                                                               00004800
       INPUT-OUTPUT SECTION.                                                   00004900
       FILE-CONTROL.                                                           00005000
                                                                               00005100
           SELECT PRINT-FILE                  ASSIGN TO DEFPR71                00005200
               FILE STATUS    PRT-STATUS.                                     00005300
                                                                               00005400
      *  REFER SELECT/ASSIGN                                                  00005500
           COPY CWAL0300.                                                      00005600
      /                                                                        00005700
```

```
 DATA DIVISION.                                                          00005800
 FILE SECTION.                                                           00005900
                                                                         00006000
 FD  PRINT-FILE                                                          00006100
     RECORDING MODE F.                                                   00006200
 01  PRT-RECORD.                                                        00006300
     05  PRT-LINE                    PIC  X(133).                       00006400
                                                                         00006500
*  REFER FILE DESCRIPTION                                               00006600
     COPY CWDL0300.                                                      00006700
                                                                         00006800
/                                                                        00006900
 WORKING-STORAGE SECTION.                                                00007000
                                                                         00007100
 01  PROGRAM-VARIABLES.                                                 00007200
   05  FILLER                           PIC X(20)  VALUE                00007300
     '=>WORKING STORAGE***'.                                             00007400
   05  WS-PROGRAM-ID                    PIC X(8)   VALUE                00007500
     'CWRB7110'.                                                         00007600
   05  WORK-PARA                        PIC X(8).                       00007700
                                                                         00007800
 01  RF-REFERENCE-RECORD.                                               00007900
     COPY CWVL0300.                                                      00008000
 01  RF-REF-RDF  REDEFINES  RF-REFERENCE-RECORD.                        00008100
   05  FILLER                            PIC X(17).                     00008200
   05  RF-FROM-DATE-X14                  PIC X(14).                     00008300
   05  FILLER                            PIC X.                         00008400
   05  RF-TO-DATE-X14                    PIC X(14).                     00008500
   05  FILLER                            PIC X(28).                     00008600
*                                                                        00008700
 01  REFER-STATUS-FLAGS.                                                00008800
     COPY CWSL0300.                                                      00008900
/                                                                        00009000
 01  RF-TABLE-ID-VALUES.                                                00009100
     COPY CWWL0388.                                                      00009200
*                                                                        00009300
 01  INPUT-PARMS.                                                       00009400
   05  IP-LOAD-DATE-RANGE-SOURCE           PIC X.                       00009500
     88  USE-REFER-FILE           VALUE  'R'.                           00009600
     88  USE-TODAYS-DATE          VALUE  'T'.                           00009700
   05  IP-FILLER1                          PIC X(39).                   00009800
/                                                                        00009900
 01  GENERAL-FIELDS.                                                    00010000
   05  WS-PGM-STATUS               PIC  X(01)  VALUE 'Y'.               00010100
         88  FIRST-TIME              VALUE 'Y'.                         00010200
         88  NOT-FIRST-TIME          VALUE 'N'.                         00010300
*                                                                        00010400
     05  REPORT-FLAG                PIC  9(01)  VALUE 0.                00010500
         88  PRT-NORM                VALUE 0.                           00010600
         88  PRT-CUST-BREAK          VALUE 1.                           00010700
         88  PRT-FINAL-TOTS          VALUE 2.                           00010800
*                                                                        00010900
   05  WS-PREV-CUST-ID              PIC  X(14).                         00011000
   05  WS-PREV-LOAD-DT-TM           PIC  X(7).                          00011100
   05  WS-PREV-LOAD-DT-TM-X14       PIC  X(14).                         00011200
   05  WS-BEN-LOAD-DT-TM-X14        PIC  X(14).                         00011300
   05  WS-PREV-LOAD-DT-TM-X17.                                          00011400
```

```
      10  WS-PLD-MM                PIC  99.                          00011500
      10  FILLER                   PIC  X   VALUE '/'.              00011600
      10  WS-PLD-DD                PIC  99.                          00011700
      10  FILLER                   PIC  X   VALUE '/'.              00011800
      10  WS-PLD-YY                PIC  99.                          00011900
      10  FILLER                   PIC  X   VALUE SPACE.            00012000
      10  WS-PLD-HH                PIC  99.                          00012100
      10  FILLER                   PIC  X   VALUE ':'.              00012200
      10  WS-PLD-MI                PIC  99.                          00012300
      10  FILLER                   PIC  X   VALUE ':'.              00012400
      10  WS-PLD-SS                PIC  99.                          00012500
*                                                                   00012600
   05  WS-BEN-LOAD-DT-CNT          PIC  S9(7)  COMP     VALUE +0.   00012700
   05  WS-BEN-CUST-TOT-CNT         PIC  S9(7)  COMP     VALUE +0.   00012800
   05  WS-BEN-REPORT-TOT-CNT       PIC  S9(7)  COMP     VALUE +0.   00012900
   05  WS-BEN-LOAD-DT-AMT          PIC  S9(7)V99 COMP-3 VALUE +0.   00013000
   05  WS-BEN-CUST-TOT-AMT         PIC  S9(7)V99 COMP-3 VALUE +0.   00013100
   05  WS-BEN-REPORT-TOT-AMT       PIC  S9(7)V99 COMP-3 VALUE +0.   00013200
*                                                                   00013300
   05  WS-FROM-DATE-X14.                                           00013400
     10  WS-FR-CC                  PIC  99.                          00013500
     10  WS-FR-YY                  PIC  99.                          00013600
     10  WS-FR-MM                  PIC  99.                          00013700
     10  WS-FR-DD                  PIC  99.                          00013800
     10  WS-FR-HH                  PIC  99.                          00013900
     10  WS-FR-MI                  PIC  99.                          00014000
     10  WS-FR-SS                  PIC  99.                          00014100
   05  WS-TO-DATE-X14.                                             00014200
     10  WS-TO-CC                  PIC  99.                          00014300
     10  WS-TO-YY                  PIC  99.                          00014400
     10  WS-TO-MM                  PIC  99.                          00014500
     10  WS-TO-DD                  PIC  99.                          00014600
     10  WS-TO-HH                  PIC  99.                          00014700
     10  WS-TO-MI                  PIC  99.                          00014800
     10  WS-TO-SS                  PIC  99.                          00014900
                                                                    00015000
/                                                                   00015100
 01  PROGRAM-FLAGS.                                                 00015200
     05  WS-PRT-OPEN-FLAG          PIC  9(01)  VALUE 0.             00015300
         88  WS-PRT-NOT-OPEN                   VALUE 0.             00015400
         88  WS-PRT-OPEN                       VALUE 1.             00015500
     05  PRT-STATUS                PIC  9(02).                      00015600
         88  PRT-SUCCESSFUL-IO                 VALUE 0.             00015700
/                                                                   00015800
 01  CWCB3000-PASS-AREA.                                            00015900
     COPY CWBL3000.                                                 00016000
/                                                                   00016100
 01  COMMON-DEFINITIONS.                                            00016200
     COPY CWWL0020.                                                 00016300
/                                                                   00016400
 01  BINARY-CONVERSION-FIELDS.                                      00016500
     COPY CWWL9430.                                                 00016600
/                                                                   00016700
 01  BATCH-RETURN-CODE.                                             00016800
     COPY CWWL9450.                                                 00016900
/                                                                   00017000
 01  WS-PROCESS-AREA.                                               00017100
```

```
     COPY CWWL0080.                                                     00017200
/                                                                       00017300
*****************************************************                   00017400
**       REPORT   DEFINITION   AREA                **                   00017500
*****************************************************                   00017600
**                                                                      00017700
 01  PRINT-CONTROL-FLDS.                                               00017800
   05  PC-SPACES                        PIC  X(133) VALUE SPACES.      00017900
   05  PC-MAX-PAGE-LINES                PIC  99  VALUE  50.            00018000
   05  PC-LINE-CNT                      PIC  99  VALUE  99.            00018100
   05  PC-PAGE-NUM                      PIC  9(4)  VALUE 1.            00018200
   05  PC-EOJ-FLAG                      PIC  X   VALUE  'N'.           00018300
     88  PC-EOJ                      VALUE  'Y'.                       00018400
   05  PLEN                             PIC  999 VALUE  ZERO.          00018500
                                                                       00018600
**                                                                      00018700
 01  HEADING-LINE1.                                                    00018800
     05  FILLER              PIC X      VALUE SPACES.                  00018900
     05  FILLER              PIC X(10)  VALUE 'RUN DATE: '.            00019000
     05  HL1-REPORT-DATE.                                              00019100
         10  HL1-RD-MM       PIC 99.                                   00019200
         10  FILLER          PIC X      VALUE '/'.                     00019300
         10  HL1-RD-DD       PIC 99.                                   00019400
         10  FILLER          PIC X      VALUE '/'.                     00019500
         10  HL1-RD-YY       PIC 99.                                   00019600
     05  FILLER              PIC X(20)  VALUE SPACES.                  00019700
     05  FILLER              PIC X(23)                                 00019800
           VALUE 'WASHINGTON METROPOLITAN'.                            00019900
     05  FILLER              PIC X(23)                                 00020000
           VALUE ' AREA TRANSIT AUTHORITY'.                            00020100
     05  FILLER              PIC X(37)  VALUE SPACES.                  00020200
     05  FILLER              PIC X(06)  VALUE 'PAGE: '.                00020300
     05  HL1-PAGE-NUMBER     PIC ZZZ9   VALUE ZEROES.                  00020400
**                                                                      00020500
 01  HEADING-LINE2.                                                    00020600
     05  FILLER              PIC X      VALUE SPACES.                  00020700
     05  FILLER              PIC X(10)  VALUE 'RUN TIME: '.            00020800
     05  HL2-RUN-TIME.                                                 00020900
       10  HL2-RT-HH         PIC 99.                                   00021000
       10  FILLER            PIC X      VALUE  ':'.                    00021100
       10  HL2-RT-MI         PIC 99.                                   00021200
       10  FILLER            PIC X      VALUE  ':'.                    00021300
       10  HL2-RT-SS         PIC 99.                                   00021400
     05  FILLER              PIC X(25)   VALUE SPACES.                 00021500
     05  FILLER              PIC X(35)                                 00021600
            VALUE 'ACCOUNTS RECEIVABLE SUMMARY REPORT'.                00021700
     05  FILLER              PIC X(39)  VALUE SPACES.                  00021800
     05  FILLER              PIC X(6)   VALUE ' PGM: '.                00021900
     05  HL2-PROGRAM-ID      PIC X(8).                                 00022000
**                                                                      00022100
************************************************************            00022200
*                CUSTOMER INFORMATION HEADINGS               *          00022300
************************************************************            00022400
*                                                                       00022500
 01  HEADING-LINE3.                                                    00022600
     05  FILLER                        PIC X      VALUE SPACES.       00022700
     05  FILLER                        PIC X(20)  VALUE SPACES.       00022800
```

```
     05  FILLER                      PIC X(17)                        00022900
                                   VALUE 'LOAD DATES FROM: '.         00023000
     05  HL3-FROM-DATE.                                               00023100
       10  HL3-FR-MM          PIC 99.                                 00023200
       10  FILLER             PIC X      VALUE '/'.                   00023300
       10  HL3-FR-DD          PIC 99.                                 00023400
       10  FILLER             PIC X      VALUE '/'.                   00023500
       10  HL3-FR-YY          PIC 99.                                 00023600
       10  FILLER             PIC X      VALUE SPACE.                 00023700
       10  HL3-FR-HH          PIC 99.                                 00023800
       10  FILLER             PIC X      VALUE  ':'.                  00023900
       10  HL3-FR-MI          PIC 99.                                 00024000
       10  FILLER             PIC X      VALUE  ':'.                  00024100
       10  HL3-FR-SS          PIC 99.                                 00024200
     05  FILLER                        PIC X(5)                       00024300
                                   VALUE ' TO: '.                     00024400
     05  HL3-TO-DATE.                                                 00024500
       10  HL3-TO-MM          PIC 99.                                 00024600
       10  FILLER             PIC X      VALUE '/'.                   00024700
       10  HL3-TO-DD          PIC 99.                                 00024800
       10  FILLER             PIC X      VALUE '/'.                   00024900
       10  HL3-TO-YY          PIC 99.                                 00025000
       10  FILLER             PIC X      VALUE SPACE.                 00025100
       10  HL3-TO-HH          PIC 99.                                 00025200
       10  FILLER             PIC X      VALUE  ':'.                  00025300
       10  HL3-TO-MI          PIC 99.                                 00025400
       10  FILLER             PIC X      VALUE  ':'.                  00025500
       10  HL3-TO-SS          PIC 99.                                 00025600
     05  FILLER                        PIC X(51)  VALUE SPACES.       00025700
*****************************************************************     00025800
*           COLUMN HEADINGS                                     *     00025900
*****************************************************************     00026000
*                                                                     00026100
 01  HEADING-LINE4.                                                   00026200
     05  FILLER                        PIC X      VALUE SPACES.       00026300
     05  FILLER                        PIC X(33)  VALUE               00026400
       '    BENEFITS LOAD DATES          '.                           00026500
     05  FILLER                        PIC X(33)  VALUE               00026600
       '# LOADED     VALUE               '.                           00026700
     05  FILLER                        PIC X(33)  VALUE               00026800
       '                                 '.                           00026900
     05  FILLER                        PIC X(33)  VALUE               00027000
       '                                 '.                           00027100
*****************************************************************     00027200
*           BANNER MESSAGES                                     *     00027300
*****************************************************************     00027400
*                                                                     00027500
 01  BAN1-L1.                                                         00027600
     05  FILLER                        PIC X      VALUE SPACES.       00027700
     05  FILLER                        PIC X(33)  VALUE               00027800
       '    ******************************'.                          00027900
     05  FILLER                        PIC X(33)  VALUE               00028000
       '**********************************'.                          00028100
     05  FILLER                        PIC X(33)  VALUE               00028200
       '**********************************'.                          00028300
     05  FILLER                        PIC X(33)  VALUE               00028400
       '                                 '.                           00028500
```

```
*                                                                       00028600
 01  BAN1-L2.                                                          00028700
     05  FILLER                      PIC X      VALUE SPACES.         00028800
     05  FILLER                      PIC X(33)  VALUE                 00028900
       '    *                             '.                          00029000
     05  FILLER                      PIC X(33)  VALUE                 00029100
       '                                  '.                          00029200
     05  FILLER                      PIC X(33)  VALUE                 00029300
       '                                 *'.                          00029400
     05  FILLER                      PIC X(33)  VALUE                 00029500
       '                                  '.                          00029600
*                                                                       00029700
 01  BAN1-L3.                                                          00029800
     05  FILLER                      PIC X      VALUE SPACES.         00029900
     05  FILLER                      PIC X(33)  VALUE                 00030000
       '                             * * * '.                          00030100
     05  FILLER                      PIC X(33)  VALUE                 00030200
       'NO BENEFITS PROCESSED * * *       '.                          00030300
     05  FILLER                      PIC X(33)  VALUE                 00030400
       '                                  '.                          00030500
     05  FILLER                      PIC X(33)  VALUE                 00030600
       '                                  '.                          00030700
*****************************************************************     00030800
*           CUSTOMER DATA LINES                                 *     00030900
*****************************************************************     00031000
*                                                                       00031100
 01  CUST-LINE1.                                                       00031200
     05  FILLER                      PIC X      VALUE SPACES.         00031300
     05  CL-HEADER                   PIC X(40)  VALUE SPACES.         00031400
     05  CL-DATA                     PIC X(60)  VALUE SPACES.         00031500
     05  FILLER                      PIC X(31)  VALUE SPACES.         00031600
*****************************************************************     00031700
*                REPORT DETAIL INFORMATION                      *     00031800
*****************************************************************     00031900
 01  DETAIL-LINE1.                                                     00032000
     05  FILLER                    PIC X      VALUE SPACES.           00032100
     05  FILLER                    PIC X(5)   VALUE SPACES.           00032200
     05  DL1-LOAD-DATE             PIC X(17).                         00032300
     05  FILLER                    PIC X(8)   VALUE SPACES.           00032400
     05  DL1-BEN-CNT               PIC ZZZ,ZZ9.                       00032500
     05  FILLER                    PIC X(5)   VALUE SPACES.           00032600
     05  DL1-BEN-AMT               PIC Z,ZZZ,ZZZ.99.                  00032700
     05  FILLER                    PIC X(77)  VALUE SPACES.           00032800
**                                                                     00032900
*****************************************************************     00033000
/                                                                      00033100
*                                                                      00033200
*    ORACLE SQLCODE CHECK VARIABLES                                    00033300
*                                                                      00033400
 01  ORA-SQLCODE-VARIABLES.                                            00033500
     COPY CWELB000.                                                    00033600
/                                                                      00033700
*****************************************************************     00033800
*    ORACLE SQL DECLARATIVES BEGIN HERE    |                      *     00033900
*                                          |                      *     00034000
*                                          V                      *     00034100
*****************************************************************     00034200
```

```
                                                                        00034300
           EXEC SQL  BEGIN DECLARE SECTION   END-EXEC.                  00034400
                                                                        00034500
      *                                                                 00034600
      *    BENEFITS TABLE DESCRIPTION                                   00034700
      *                                                                 00034800
-INC CWELB100                                                           0003490%
      *                                                                 00035000
      *    BENEFITS TABLE EQUIVALENCIES                                 00035100
      *                                                                 00035200
-INC CWELB101                                                           0003530%
      /                                                                 00035400
      *                                                                 00035500
      *    CUSTOMER TABLE DESCRIPTION                                   00035600
      *                                                                 00035700
-INC CWELB300                                                           0003580%
      *                                                                 00035900
      *    CUSTOMER TABLE EQUIVALENCIES                                 00036000
      *                                                                 00036100
-INC CWELB301                                                           0003620%
      *                                                                 00036300
       01  HOST-VARIABLES.                                             00036400
         05  WS-FROM-DATE                  PIC X(7).                   00036500
         05  WS-TO-DATE                    PIC X(7).                   00036600
      *                                                                 00036700
         05  WS-PROCESS-DATE-X14              PIC   X(14).              00036800
         05  WS-PD-X14  REDEFINES WS-PROCESS-DATE-X14.                  00036900
           10  WS-PD-DATE.                                             00037000
             15  WS-PD-CC                        PIC  S99.           00037100
             15  WS-PD-YY                        PIC  S99.           00037200
             15  WS-PD-MM                        PIC  S99.           00037300
             15  WS-PD-DD                        PIC  S99.           00037400
           10  WS-PD-TIME.                                             00037500
             15  WS-PD-HH                        PIC  S99.           00037600
             15  WS-PD-MI                        PIC  S99.           00037700
             15  WS-PD-SS                        PIC  S99.           00037800
      *                                                                 00037900
           EXEC SQL VAR                                                 00038000
             WS-FROM-DATE            IS  DATE                         00038100
           END-EXEC                                                     00038200
      *                                                                 00038300
           EXEC SQL VAR                                                 00038400
             WS-TO-DATE              IS  DATE                         00038500
           END-EXEC                                                     00038600
      *                                                                 00038700
           EXEC SQL  END DECLARE SECTION    END-EXEC                    00038800
      *                                                                 00038900
      /                                                                 00039000
           EXEC SQL  INCLUDE SQLCA                                      00039100
               INCLUDE SQLCA                                            00039200
           END-EXEC                                                     00039300
      *                                                                 00039400
      *                                                                 00039500
      *    BENEFITS CURSOR                                              00039600
      *                                                                 00039700
           EXEC SQL   DECLARE  BENEFITS_LOAD  CURSOR FOR                00039800
             SELECT                                                     00039900
```

```
-INC CWELB800                                                           0004000%
             FROM  MCHECK.BENEFITS                                      00040100
              WHERE      LOAD_DT_TM    >=   :WS-FROM-DATE              00040200
               AND       LOAD_DT_TM    <=   :WS-TO-DATE                00040300
               ORDER BY  CUSTOMER_ID,                                   00040400
                         LOAD_DT_TM                                     00040500
          END-EXEC                                                      00040600
        /                                                               00040700
         PROCEDURE DIVISION.                                            00040800
         A000-MAIN-ROUTINE.                                             00040900
        *     DISPLAY 'A000: START'                                     00041000
              PERFORM B000-INITIALIZE THRU                              00041100
                      B000-INITIALIZE-EXIT                              00041200
                                                                        00041300
              IF  BRC-IN-PROCESS                                        00041400
                PERFORM  C000-PROCESS-CUSTOMER THRU                     00041500
                         C000-PROCESS-CUSTOMER-EXIT                     00041600
                  UNTIL  BRC-STOP-PROCESSING                            00041700
              END-IF                                                    00041800
                                                                        00041900
              PERFORM  E000-CLEAN-UP THRU                               00042000
                       E000-CLEAN-UP-EXIT                               00042100
              .                                                         00042200
         A000-MAIN-ROUTINE-EXIT.                                        00042300
              GOBACK.                                                   00042400
        /                                                               00042500
         B000-INITIALIZE.                                               00042600
              DISPLAY 'B000: START'                                     00042700
                                                                        00042800
              SET BRC-IN-PROCESS      TO      TRUE                      00042900
                                                                        00043000
              OPEN OUTPUT  PRINT-FILE                                   00043100
              IF  NOT PRT-SUCCESSFUL-IO                                 00043200
                DISPLAY 'ERROR OPENING PRINT-FILE  : ' PRT-STATUS       00043300
                SET BRC-OPEN-ERROR  TO  TRUE                            00043400
                GO TO B000-INITIALIZE-EXIT                              00043500
              END-IF                                                    00043600
        *                                                               00043700
        *    REFER FILE HAS DATE RANGE PARMETERS UNTIL SOMETHING        00043800
        *       ELSE COMES ALONG                                        00043900
        *                                                               00044000
              ACCEPT  INPUT-PARMS                                       00044100
              DISPLAY 'INPUT PARMS: '  INPUT-PARMS                     00044200
        *                                                               00044300
              IF  USE-TODAYS-DATE                                       00044400
                DISPLAY  'TODAYS DATE'                                  00044500
                PERFORM  B130-GET-TODAYS-RANGE THRU                     00044600
                         B130-GET-TODAYS-RANGE-EXIT                     00044700
              ELSE                                                      00044800
                PERFORM  B150-GET-ENTERED-RANGE THRU                    00044900
                         B150-GET-ENTERED-RANGE-EXIT                    00045000
              END-IF                                                    00045100
        *                                                               00045200
              EXEC SQL  SELECT TO_CHAR(SYSDATE, 'YYYYMMDDHH24MISS')     00045300
                 INTO  :WS-PROCESS-DATE-X14                             00045400
                FROM  DUAL                                              00045500
              END-EXEC                                                  00045600
```

```
     DISPLAY 'PROCESS-DATE: '    WS-PROCESS-DATE-X14                  00045700
*                                                                     00045800
     MOVE  PC-PAGE-NUM                  TO  HL1-PAGE-NUMBER           00045900
     MOVE  WS-PD-MM                     TO  HL1-RD-MM                 00046000
     MOVE  WS-PD-DD                     TO  HL1-RD-DD                 00046100
     MOVE  WS-PD-YY                     TO  HL1-RD-YY                 00046200
     MOVE  WS-PD-HH                     TO  HL2-RT-HH                 00046300
     MOVE  WS-PD-MI                     TO  HL2-RT-MI                 00046400
     MOVE  WS-PD-SS                     TO  HL2-RT-SS                 00046500
     MOVE  WS-PROGRAM-ID                TO  HL2-PROGRAM-ID            00046600
     MOVE  WS-FR-MM                     TO  HL3-FR-MM                 00046700
     MOVE  WS-FR-DD                     TO  HL3-FR-DD                 00046800
     MOVE  WS-FR-YY                     TO  HL3-FR-YY                 00046900
     MOVE  WS-FR-HH                     TO  HL3-FR-HH                 00047000
     MOVE  WS-FR-MI                     TO  HL3-FR-MI                 00047100
     MOVE  WS-FR-SS                     TO  HL3-FR-SS                 00047200
     MOVE  WS-TO-MM                     TO  HL3-TO-MM                 00047300
     MOVE  WS-TO-DD                     TO  HL3-TO-DD                 00047400
     MOVE  WS-TO-YY                     TO  HL3-TO-YY                 00047500
     MOVE  WS-TO-HH                     TO  HL3-TO-HH                 00047600
     MOVE  WS-TO-MI                     TO  HL3-TO-MI                 00047700
     MOVE  WS-TO-SS                     TO  HL3-TO-SS                 00047800
*                                                                     00047900
*    OPEN BENEFITS LOAD DATE RANGE CURSOR                             00048000
*                                                                     00048100
     PERFORM  V100-OPEN-BEN-LOAD THRU                                 00048200
              V100-OPEN-BEN-LOAD-EXIT                                 00048300
     .                                                               00048400
 B000-INITIALIZE-EXIT.                                               00048500
     EXIT.                                                           00048600
/                                                                    00048700
 B130-GET-TODAYS-RANGE.                                              00048800
     DISPLAY 'B130: START'                                           00048900
*                                                                    00049000
     MOVE '0'                          TO BDT-PROCESS-DIR            00049100
     PERFORM D000-GMTCONV-PROCESS THRU                               00049200
             D000-GMTCONV-PROCESS-EXIT                               00049300
     IF  BRC-STOP-PROCESSING                                         00049400
       GO TO  B130-GET-TODAYS-RANGE-EXIT                             00049500
     END-IF                                                          00049600
                                                                     00049700
     MOVE  BDT-DATE                TO  WS-FROM-DATE-X14              00049800
                                       WS-TO-DATE-X14                00049900
     MOVE  ZEROES                  TO  WS-FROM-DATE-X14 (9 : 6)      00050000
     MOVE  '235959'                TO  WS-TO-DATE-X14   (9 : 6)      00050100
     DISPLAY 'TODAYS DATE RANGE FROM: '   WS-FROM-DATE-X14            00050200
             ' TO: '                      WS-TO-DATE-X14              00050300
                                                                     00050400
     MOVE  WS-FROM-DATE-X14        TO  WS-DT-TM-X                    00050500
     PERFORM D180-DATE-X14-TO-ORA THRU                               00050600
             D180-DATE-X14-TO-ORA-EXIT                               00050700
     MOVE  WS-ORA-DT-TM            TO  WS-FROM-DATE                  00050800
                                                                     00050900
     MOVE  WS-TO-DATE-X14        TO  WS-DT-TM-X                      00051000
     PERFORM D180-DATE-X14-TO-ORA THRU                               00051100
             D180-DATE-X14-TO-ORA-EXIT                               00051200
     MOVE  WS-ORA-DT-TM            TO  WS-TO-DATE                    00051300
```

```
           .                                                            00051400
        B130-GET-TODAYS-RANGE-EXIT.                                     00051500
            EXIT.                                                       00051600
        *                                                               00051700
        B150-GET-ENTERED-RANGE.                                         00051800
            DISPLAY 'B150: START'                                       00051900
                                                                        00052000
            OPEN INPUT REFER-FILE                                       00052100
            IF  RF-SUCCESSFUL-IO                                        00052200
              SET  RF-REFER-OPEN TO TRUE                                00052300
            ELSE                                                        00052400
              DISPLAY '*** ERROR OPENING REFERENCE FILE *** '           00052500
                         'STATUS: ' RF-STATUS                           00052600
              SET  BRC-OPEN-ERROR  TO  TRUE                             00052700
              GO TO B150-GET-ENTERED-RANGE-EXIT                         00052800
            END-IF                                                      00052900
                                                                        00053000
            SET  RF-DATE-RANGE-SELECTION TO  TRUE                       00053100
            MOVE REFERENCE-FILE-VALUES    TO  RF-TABLE-ID               00053200
            MOVE WS-PROGRAM-ID            TO  RF-TABLE-ENTRY-ID         00053300
            MOVE RF-KEY                   TO  RF-PRIME-KEY              00053400
            READ REFER-FILE INTO RF-REFERENCE-RECORD                    00053500
                        KEY IS RF-PRIME-KEY                             00053600
                                                                        00053700
            IF  NOT RF-SUCCESSFUL-IO                                    00053800
              SET  BRC-IO-ERROR  TO TRUE                                00053900
              DISPLAY '*** ERROR READING REFERENCE FILE *** '           00054000
                     'STATUS: ' RF-STATUS                               00054100
              GO TO B150-GET-ENTERED-RANGE-EXIT                         00054200
            END-IF                                                      00054300
                                                                        00054400
        DEBUGX      DISPLAY 'REFERENCE RECORD: '  RF-REFERENCE-RECORD   00054500
        DEBUGX      DISPLAY  'REFERENCE FILE DATES: '                   00054600
        DEBUGX      DISPLAY  'FROM DATE: '           RF-FROM-DATE-X14   00054700
        DEBUGX               ' TO DATE: '            RF-TO-DATE-X14     00054800
                                                                        00054900
            MOVE  RF-FROM-DATE-X14         TO  BDT-DATE                 00055000
            MOVE  RF-FROM-DATE-X14 (9 : 6) TO  BDT-TIME                 00055100
                                                                        00055200
            MOVE '1'                      TO BDT-PROCESS-DIR            00055300
            PERFORM D000-GMTCONV-PROCESS THRU                           00055400
                    D000-GMTCONV-PROCESS-EXIT                           00055500
            IF  BRC-STOP-PROCESSING                                     00055600
              GO TO B150-GET-ENTERED-RANGE-EXIT                         00055700
            END-IF                                                      00055800
                                                                        00055900
            MOVE  RF-TO-DATE-X14           TO  BDT-DATE                 00056000
            MOVE  RF-TO-DATE-X14 (9 : 6)   TO  BDT-TIME                 00056100
                                                                        00056200
            MOVE '1'                      TO BDT-PROCESS-DIR            00056300
            PERFORM D000-GMTCONV-PROCESS THRU                           00056400
                    D000-GMTCONV-PROCESS-EXIT                           00056500
            IF  BRC-STOP-PROCESSING                                     00056600
              GO TO B150-GET-ENTERED-RANGE-EXIT                         00056700
            END-IF                                                      00056800
                                                                        00056900
            MOVE  RF-FROM-DATE-X14        TO  WS-DT-TM-X                00057000
```

```
                                          WS-FROM-DATE-X14           00057100
           PERFORM D180-DATE-X14-TO-ORA THRU                         00057200
                   D180-DATE-X14-TO-ORA-EXIT                         00057300
           MOVE  WS-ORA-DT-TM            TO  WS-FROM-DATE            00057400
                                                                     00057500
           MOVE  RF-TO-DATE-X14          TO  WS-DT-TM-X              00057600
                                             WS-TO-DATE-X14          00057700
           PERFORM D180-DATE-X14-TO-ORA THRU                         00057800
                   D180-DATE-X14-TO-ORA-EXIT                         00057900
           MOVE  WS-ORA-DT-TM            TO  WS-TO-DATE              00058000
           .                                                         00058100
       B150-GET-ENTERED-RANGE-EXIT.                                  00058200
           EXIT.                                                     00058300
      /                                                              00058400
       C000-PROCESS-CUSTOMER.                                        00058500
           DISPLAY 'C000: START'                                     00058600
                                                                     00058700
      *                                                              00058800
      *    FETCH BENEFITS BY LOAD DATE/TIME                          00058900
      *                                                              00059000
           PERFORM  V150-FETCH-BENEFITS-LOAD THRU                    00059100
                    V150-FETCH-BENEFITS-LOAD-EXIT                    00059200
                                                                     00059300
DEBUGX     DISPLAY  'BENEFITS ROW: '  BENEFITS-ROW                  00059400
           EVALUATE TRUE                                             00059500
             WHEN ORA-SQL-SUCCESSFUL                                 00059600
               IF FIRST-TIME                                         00059700
                 MOVE  BEN-LOAD-DT-TM              TO  WS-ORA-DT-TM  00059800
                 PERFORM D100-DATE-ORA-X14 THRU                      00059900
                         D100-DATE-ORA-X14-EXIT                      00060000
                 MOVE  WS-DT-TM-X          TO  WS-BEN-LOAD-DT-TM-X14 00060100
                 SET   NOT-FIRST-TIME  TO  TRUE                      00060200
                 MOVE BEN-CUSTOMER-ID              TO  WS-PREV-CUST-ID 00060300
                 MOVE WS-BEN-LOAD-DT-TM-X14                          00060400
                                          TO  WS-PREV-LOAD-DT-TM-X14 00060500
                 PERFORM P500-HEADINGS THRU                          00060600
                         P500-HEADINGS-EXIT                          00060700
               END-IF                                                00060800
                                                                     00060900
               IF  BRC-IN-PROCESS                                    00061000
                 PERFORM  C100-SUM-BENEFITS THRU                     00061100
                          C100-SUM-BENEFITS-EXIT                     00061200
               END-IF                                                00061300
                                                                     00061400
             WHEN ORA-SQL-ROW-NOT-FOUND                              00061500
               SET BRC-SUCCESSFUL-COMPLETION  TO TRUE                00061600
                                                                     00061700
             WHEN OTHER                                              00061800
               SET BRC-SQL-ERROR  TO  TRUE                           00061900
               DISPLAY                                               00062000
                   'I/O ERROR - BENEFITS LOAD CURSOR  '              00062100
           END-EVALUATE                                              00062200
      *    SET  BRC-SUCCESSFUL-COMPLETION  TO  TRUE                  00062300
           .                                                         00062400
       C000-PROCESS-CUSTOMER-EXIT.                                   00062500
           EXIT.                                                     00062600
      /                                                              00062700
```

```
       C100-SUM-BENEFITS.                                                       00062800
      *                                                                         00062900
           DISPLAY  'C100: '                                                    00063000
                                                                                00063100
           MOVE  BEN-LOAD-DT-TM              TO  WS-ORA-DT-TM                   00063200
           PERFORM D100-DATE-ORA-X14 THRU                                       00063300
                   D100-DATE-ORA-X14-EXIT                                       00063400
           MOVE  WS-DT-TM-X                  TO  WS-BEN-LOAD-DT-TM-X14          00063500
DEBUGX     DISPLAY  'C100: PREV LOAD DT: '  WS-PREV-LOAD-DT-TM-X14              00063600
DEBUGX              ' BEN LOAD DT: '  WS-BEN-LOAD-DT-TM-X14                     00063700
                                                                                00063800
           EVALUATE  TRUE                                                       00063900
             WHEN  BEN-CUSTOMER-ID  NOT =  WS-PREV-CUST-ID                      00064000
DEBUGX     DISPLAY  'C100: CUST BREAK'                                          00064100
               PERFORM  P100-PRT-LOAD-DT-TOTAL THRU                             00064200
                        P100-PRT-LOAD-DT-TOTAL-EXIT                             00064300
               PERFORM  P300-PRT-CUST-TOTAL THRU                                00064400
                        P300-PRT-CUST-TOTAL-EXIT                                00064500
               MOVE BEN-CUSTOMER-ID              TO  WS-PREV-CUST-ID           00064600
               PERFORM  P500-HEADINGS THRU                                      00064700
                        P500-HEADINGS-EXIT                                      00064800
      *        MOVE BEN-LOAD-DT-TM               TO  WS-PREV-LOAD-DT-TM        00064900
               MOVE WS-BEN-LOAD-DT-TM-X14  TO  WS-PREV-LOAD-DT-TM-X14           00065000
                                                                                00065100
             WHEN  WS-BEN-LOAD-DT-TM-X14                                        00065200
                                   NOT =  WS-PREV-LOAD-DT-TM-X14                00065300
DEBUGX     DISPLAY  'C100: LOAD DT BREAK'                                       00065400
               PERFORM  P100-PRT-LOAD-DT-TOTAL THRU                             00065500
                        P100-PRT-LOAD-DT-TOTAL-EXIT                             00065600
      *        MOVE BEN-LOAD-DT-TM               TO  WS-PREV-LOAD-DT-TM        00065700
               MOVE WS-BEN-LOAD-DT-TM-X14  TO  WS-PREV-LOAD-DT-TM-X14           00065800
                                                                                00065900
           END-EVALUATE                                                         00066000
                                                                                00066100
           IF  BRC-IN-PROCESS                                                   00066200
DEBUGX     DISPLAY  'C100: ADD AMOUNTS  '                                       00066300
             ADD   1   TO  WS-BEN-LOAD-DT-CNT                                   00066400
             ADD  BEN-INITIAL-VAL-AMT  TO  WS-BEN-LOAD-DT-AMT                   00066500
           END-IF                                                               00066600
           .                                                                    00066700
       C100-SUM-BENEFITS-EXIT.                                                  00066800
           EXIT.                                                                00066900
      /                                                                         00067000
       D000-GMTCONV-PROCESS.                                                    00067100
      *                                                                         00067200
           CALL 'CWCB3000' USING  CWCB3000-PASS-AREA                            00067300
                                                                                00067400
           IF BDT-RTN-CODE > 0                                                  00067500
             MOVE  BDT-RTN-CODE                TO  BRC-RETURN-CODE            00067600
             DISPLAY ' BDT-RTN-CODE = ' BDT-RTN-CODE                            00067700
             DISPLAY ' PASS AREA = ' CWCB3000-PASS-AREA                         00067800
           END-IF                                                               00067900
           .                                                                    00068000
       D000-GMTCONV-PROCESS-EXIT.                                               00068100
           EXIT.                                                                00068200
                                                                                00068300
      ***************************************************************************00068400
```

```
*  CONVERTS AN ORACLE DATE/TIME TO PIC X14 CCYYMMDDHHMMSS        *00068500
*****************************************************************00068600
 D100-DATE-ORA-X14.                                              00068700
                                                                 00068800
     MOVE  WS-ODT-BYTE (1)          TO  WS-B2-LO                 00068900
     SUBTRACT  100  FROM  WS-BIN-2N                              00069000
     MOVE  WS-BIN-2N                TO  WS-DT-CC                 00069100
     MOVE  WS-ODT-BYTE (2)          TO  WS-B2-LO                 00069200
     SUBTRACT  100  FROM  WS-BIN-2N                              00069300
     MOVE  WS-BIN-2N                TO  WS-DT-YY                 00069400
     MOVE  WS-ODT-BYTE (3)          TO  WS-B2-LO                 00069500
     MOVE  WS-BIN-2N                TO  WS-DT-MM                 00069600
     MOVE  WS-ODT-BYTE (4)          TO  WS-B2-LO                 00069700
     MOVE  WS-BIN-2N                TO  WS-DT-DD                 00069800
     MOVE  WS-ODT-BYTE (5)          TO  WS-B2-LO                 00069900
     SUBTRACT  1    FROM  WS-BIN-2N                              00070000
     MOVE  WS-BIN-2N                TO  WS-DT-HH                 00070100
     MOVE  WS-ODT-BYTE (6)          TO  WS-B2-LO                 00070200
     SUBTRACT  1    FROM  WS-BIN-2N                              00070300
     MOVE  WS-BIN-2N                TO  WS-DT-MI                 00070400
     MOVE  WS-ODT-BYTE (7)          TO  WS-B2-LO                 00070500
     SUBTRACT  1    FROM  WS-BIN-2N                              00070600
     MOVE  WS-BIN-2N                TO  WS-DT-SS                 00070700
     .                                                           00070800
 D100-DATE-ORA-X14-EXIT.                                         00070900
     EXIT.                                                       00071000
                                                                 00071100
*****************************************************************00071200
*****************************************************************00071300
 D180-DATE-X14-TO-ORA.                                           00071400
     MOVE  WS-DT-CC                 TO  WS-BIN-2N                00071500
     ADD       100  TO    WS-BIN-2N                             00071600
     MOVE  WS-B2-LO                 TO  WS-ODT-BYTE (1)          00071700
     MOVE  WS-DT-YY                 TO  WS-BIN-2N                00071800
     ADD       100  TO    WS-BIN-2N                             00071900
     MOVE  WS-B2-LO                 TO  WS-ODT-BYTE (2)          00072000
     MOVE  WS-DT-MM                 TO  WS-BIN-2N                00072100
     MOVE  WS-B2-LO                 TO  WS-ODT-BYTE (3)          00072200
     MOVE  WS-DT-DD                 TO  WS-BIN-2N                00072300
     MOVE  WS-B2-LO                 TO  WS-ODT-BYTE (4)          00072400
     MOVE  WS-DT-HH                 TO  WS-BIN-2N                00072500
     ADD       1    TO    WS-BIN-2N                             00072600
     MOVE  WS-B2-LO                 TO  WS-ODT-BYTE (5)          00072700
     MOVE  WS-DT-MI                 TO  WS-BIN-2N                00072800
     ADD       1    TO    WS-BIN-2N                             00072900
     MOVE  WS-B2-LO                 TO  WS-ODT-BYTE (6)          00073000
     MOVE  WS-DT-SS                 TO  WS-BIN-2N                00073100
     ADD       1    TO    WS-BIN-2N                             00073200
     MOVE  WS-B2-LO                 TO  WS-ODT-BYTE (7)          00073300
     .                                                           00073400
 D180-DATE-X14-TO-ORA-EXIT.                                      00073500
     EXIT.                                                       00073600
/                                                                00073700
 E000-CLEAN-UP.                                                  00073800
*****************************************************************00073900
* THIS PROCESS CLOSES FILES AND PERFORMS OTHER POST-PROCESSING   *00074000
* TASKS                                                         *00074100
```

```
      ***************************************************************** 00074200
                                                                        00074300
      *                                                                 00074400
           SET  PC-EOJ  TO  TRUE                                        00074500
                                                                        00074510
                                                                        00074540
           IF  NOT FIRST-TIME                                           00074600
             PERFORM  P100-PRT-LOAD-DT-TOTAL THRU                       00074700
                      P100-PRT-LOAD-DT-TOTAL-EXIT                       00074800
             PERFORM  P300-PRT-CUST-TOTAL THRU                          00074900
                      P300-PRT-CUST-TOTAL-EXIT                          00075000
             PERFORM  P500-HEADINGS THRU                                00074520
                      P500-HEADINGS-EXIT                                00074530
           ELSE                                                         00075400
             PERFORM  P500-HEADINGS THRU                                00074520
                      P500-HEADINGS-EXIT                                00074530
             PERFORM  P550-NO-PROC-MSG THRU                             00075200
                      P550-NO-PROC-MSG-EXIT                             00075300
           END-IF                                                       00075400
                                                                        00075500
           PERFORM  P400-PRT-REPORT-TOTAL THRU                          00075800
                    P400-PRT-REPORT-TOTAL-EXIT                          00075900
      *                                                                 00076000
      *    CLOSE FILES                                                  00076100
      *                                                                 00076200
           CLOSE PRINT-FILE                                             00076300
           .                                                            00076400
       E000-CLEAN-UP-EXIT.                                              00076500
           EXIT.                                                        00076600
      /                                                                 00076700
       P100-PRT-LOAD-DT-TOTAL.                                          00076800
DEBUGX     DISPLAY 'P100:   START'                                      00076900
                                                                        00077000
      *    MOVE  WS-PREV-LOAD-DT-TM          TO  WS-ORA-DT-TM          00077100
      *    PERFORM D100-DATE-ORA-X14 THRU                              00077200
      *            D100-DATE-ORA-X14-EXIT                              00077300
      *    MOVE  WS-DT-TM-X                  TO  WS-PROCESS-DATE-X14   00077400
           MOVE  WS-PREV-LOAD-DT-TM-X14      TO  WS-PROCESS-DATE-X14   00077500
                                                                        00077600
           MOVE  WS-PD-MM                    TO  WS-PLD-MM             00077700
           MOVE  WS-PD-DD                    TO  WS-PLD-DD             00077800
           MOVE  WS-PD-YY                    TO  WS-PLD-YY             00077900
           MOVE  WS-PD-HH                    TO  WS-PLD-HH             00078000
           MOVE  WS-PD-MI                    TO  WS-PLD-MI             00078100
           MOVE  WS-PD-SS                    TO  WS-PLD-SS             00078200
           MOVE  WS-PREV-LOAD-DT-TM-X17      TO  DL1-LOAD-DATE         00078300
                                                                        00078400
           MOVE  WS-BEN-LOAD-DT-CNT          TO  DL1-BEN-CNT           00078500
           MOVE  WS-BEN-LOAD-DT-AMT          TO  DL1-BEN-AMT           00078600
           WRITE PRT-RECORD  FROM  DETAIL-LINE1 AFTER 1                 00078700
                                                                        00078800
           ADD  WS-BEN-LOAD-DT-CNT  TO  WS-BEN-CUST-TOT-CNT             00078900
           ADD  WS-BEN-LOAD-DT-AMT  TO  WS-BEN-CUST-TOT-AMT             00079000
           MOVE ZERO                         TO  WS-BEN-LOAD-DT-CNT     00079100
           MOVE ZERO                         TO  WS-BEN-LOAD-DT-AMT     00079200
           .                                                            00079300
       P100-PRT-LOAD-DT-TOTAL-EXIT.                                     00079400
```

```
           EXIT.                                                        00079500
       /                                                                00079600
        P300-PRT-CUST-TOTAL.                                            00079700
DEBUGX     DISPLAY 'P300:   START'                                      00079800
                                                                        00079900
           MOVE  'TOTAL LOADED     '          TO  DL1-LOAD-DATE         00080000
           MOVE  WS-BEN-CUST-TOT-CNT          TO  DL1-BEN-CNT           00080100
           MOVE  WS-BEN-CUST-TOT-AMT          TO  DL1-BEN-AMT           00080200
           WRITE PRT-RECORD  FROM  DETAIL-LINE1 AFTER 1                 00080300
                                                                        00080400
           ADD  WS-BEN-CUST-TOT-CNT TO  WS-BEN-REPORT-TOT-CNT           00080500
           ADD  WS-BEN-CUST-TOT-AMT TO  WS-BEN-REPORT-TOT-AMT           00080600
           MOVE ZERO                        TO  WS-BEN-CUST-TOT-CNT     00080700
           MOVE ZERO                        TO  WS-BEN-CUST-TOT-AMT     00080800
           .                                                            00080900
        P300-PRT-CUST-TOTAL-EXIT.                                       00081000
           EXIT.                                                        00081100
       /                                                                00081200
        P400-PRT-REPORT-TOTAL.                                          00081300
DEBUGX     DISPLAY 'P400:   START'                                      00081400
                                                                        00081500
           MOVE  'REPORT TOTALS    '          TO  DL1-LOAD-DATE         00081600
           MOVE  WS-BEN-REPORT-TOT-CNT          TO  DL1-BEN-CNT         00081700
           MOVE  WS-BEN-REPORT-TOT-AMT          TO  DL1-BEN-AMT         00081800
           WRITE PRT-RECORD  FROM  DETAIL-LINE1 AFTER 1                 00081900
           .                                                            00082000
        P400-PRT-REPORT-TOTAL-EXIT.                                     00082100
           EXIT.                                                        00082200
       /                                                                00082300
        P500-HEADINGS.                                                  00082400
                                                                        00082500
       *    DISPLAY 'P500:  START'                                      00082600
           MOVE    'P500:   '                TO  WORK-PARA             00082700
       *                                                                00082800
           WRITE PRT-RECORD  FROM  HEADING-LINE1  AFTER TOP-OF-PAGE     00082900
           WRITE PRT-RECORD  FROM  HEADING-LINE2 AFTER 1                00083000
           WRITE PRT-RECORD  FROM  HEADING-LINE3 AFTER 2                00083100
           IF  NOT  PC-EOJ                                              00083200
             PERFORM  P530-PRINT-CUST-DATA THRU                         00083300
                      P530-PRINT-CUST-DATA-EXIT                         00083400
           END-IF                                                       00083500
           WRITE PRT-RECORD  FROM  HEADING-LINE4 AFTER 2                00083600
           WRITE PRT-RECORD  FROM  PC-SPACES     AFTER 1                00083700
           ADD  1  TO  PC-PAGE-NUM                                      00083800
           MOVE  PC-PAGE-NUM                TO  HL1-PAGE-NUMBER         00083900
           MOVE    6                        TO  PC-LINE-CNT             00084000
           .                                                            00084100
        P500-HEADINGS-EXIT.                                             00084200
            EXIT.                                                       00084300
       /                                                                00084400
       ****************************************************************00084500
       *  SELECT CUSTOMER ROW AND PRINTS CUSTOMER DATA                 *00084600
       ****************************************************************00084700
        P530-PRINT-CUST-DATA.                                           00084800
       *    DISPLAY 'P530:  START'                                      00084900
                                                                        00085000
           PERFORM  V200-SELECT-CUSTOMER THRU                           00085100
```

```
                V200-SELECT-CUSTOMER-EXIT                                00085200
                                                                         00085300
           EVALUATE TRUE                                                 00085400
             WHEN ORA-SQL-SUCCESSFUL                                     00085500
DEBUGX     DISPLAY  'CUSTOMER ROW: '  CUSTOMER-ROW                       00085600
                                                                         00085700
              MOVE  'CUSTOMER ID          '      TO  CL-HEADER          00085800
              MOVE  CUS-CUSTOMER-ID              TO  CL-DATA            00085900
              WRITE PRT-RECORD  FROM  CUST-LINE1 AFTER 1                 00086000
                                                                         00086100
       * * * *   CUSTOMER  NAME                                          00086200
              MOVE  'CUSTOMER NAME        '      TO  CL-HEADER          00086300
              IF    CUS-CUSTOMER-NAME-LEN  >  ZERO                       00086400
                MOVE  CUS-CUSTOMER-NAME-ARR      TO  CL-DATA            00086500
              ELSE                                                       00086600
                MOVE  SPACES                     TO  CL-DATA            00086700
              END-IF                                                     00086800
              WRITE PRT-RECORD  FROM  CUST-LINE1 AFTER 1                 00086900
                                                                         00087000
       * * * *   STREET ADDRESS 1                                       00087100
              MOVE  'ADDRESS 1            '      TO  CL-HEADER          00087200
              IF    CUS-STREET-ADDR1-LEN   >  ZERO                       00087300
                MOVE  CUS-STREET-ADDR1-ARR       TO  CL-DATA            00087400
              ELSE                                                       00087500
                MOVE  SPACES                     TO  CL-DATA            00087600
              END-IF                                                     00087700
              WRITE PRT-RECORD  FROM  CUST-LINE1 AFTER 1                 00087800
                                                                         00087900
       * * * *   STREET ADDRESS 2                                       00088000
              MOVE  'ADDRESS 2            '      TO  CL-HEADER          00088100
              IF    CUS-STREET-ADDR2-LEN   >  ZERO                       00088200
                MOVE  CUS-STREET-ADDR2-ARR       TO  CL-DATA            00088300
              ELSE                                                       00088400
                MOVE  SPACES                     TO  CL-DATA            00088500
              END-IF                                                     00088600
              WRITE PRT-RECORD  FROM  CUST-LINE1 AFTER 1                 00088700
                                                                         00088800
       * * * *   CITY, STATE ZIP                                        00088900
              MOVE  SPACES                       TO  CL-DATA            00089000
              MOVE  'CITY, STATE ZIP      '      TO  CL-HEADER          00089100
              MOVE  SPACES                       TO  CL-DATA            00089200
              MOVE  CUS-CITY-LEN                 TO  PLEN               00089300
              IF  PLEN  >  ZERO                                          00089400
                MOVE  CUS-CITY-ARR          TO  CL-DATA (1 : PLEN)       00089500
                ADD  1  TO  PLEN                                         00089600
                MOVE   ', '                 TO  CL-DATA (PLEN : 2)       00089700
                ADD  2  TO  PLEN                                         00089800
              ELSE                                                       00089900
                MOVE   1                    TO  PLEN                     00090000
              END-IF                                                     00090100
                                                                         00090200
              IF  CUS-STATE-LEN   >  ZERO                                00090300
                MOVE  CUS-STATE-ARR         TO  CL-DATA (PLEN : 2)       00090400
                ADD  3  TO  PLEN                                         00090500
              END-IF                                                     00090600
                                                                         00090700
              IF  CUS-ZIPCODE-LEN   >  ZERO                              00090800
```

```
          MOVE  CUS-ZIPCODE-ARR       TO  CL-DATA (PLEN : 5)       00090900
          ADD  5  TO  PLEN                                         00091000
        END-IF                                                     00091100
                                                                   00091200
        IF  CUS-ZIPCODE-4-LEN   >  ZERO                            00091300
          MOVE  '-' TO  CL-DATA (PLEN : 1)                         00091400
          ADD  1  TO  PLEN                                         00091500
          MOVE  CUS-ZIPCODE-4-ARR     TO  CL-DATA (PLEN : 4)       00091600
        END-IF                                                     00091700
        WRITE PRT-RECORD  FROM  CUST-LINE1 AFTER 1                 00091800
                                                                   00091900
* * * *   CONTRACT NUMBER, PO NUMBER ?????                         00092000
        MOVE  'ACCOUNTS RECEIVABLE MISCELLANEOUS DATA: '           00092100
                                              TO  CL-HEADER        00092200
        IF    CUS-ACCTS-RECV-MISC-DATA-LEN  >  ZERO                00092300
          MOVE  CUS-ACCTS-RECV-MISC-DATA-ARR  TO  CL-DATA          00092400
        ELSE                                                       00092500
          MOVE  SPACES                        TO  CL-DATA          00092600
        END-IF                                                     00092700
        WRITE PRT-RECORD  FROM  CUST-LINE1 AFTER 2                 00092800
                                                                   00092900
* * * *   USER ID OF CONTACT PERSON W/ TITLE                       00093000
        MOVE  'CONTACT NAME, TITLE     '   TO  CL-HEADER           00093100
        MOVE  CUS-CONTACT-USER-ID-LEN      TO  PLEN                00093200
        IF  PLEN  >  ZERO                                          00093300
          MOVE  CUS-CONTACT-USER-ID-ARR    TO  CL-DATA             00093400
        ELSE                                                       00093500
          MOVE  SPACES                     TO  CL-DATA             00093600
        END-IF                                                     00093700
                                                                   00093800
        IF  CUS-CONTACT-TITLE-LEN   >  ZERO                        00093900
          ADD  1  TO  PLEN                                         00094000
          MOVE   ', '                    TO  CL-DATA (PLEN : 2)    00094100
          ADD  3  TO  PLEN                                         00094200
          MOVE  CUS-CONTACT-TITLE-ARR     TO  CL-DATA (PLEN : 20)  00094300
        END-IF                                                     00094400
        WRITE PRT-RECORD  FROM  CUST-LINE1 AFTER 1                 00094500
                                                                   00094600
* * * *   TELEPHONE NUMBER OF CUSTOMER                             00094700
        MOVE  'PHONE NUMBER          '     TO  CL-HEADER           00094800
        IF    CUS-PHONE-NUMBER-LEN   >  ZERO                       00094900
          MOVE  CUS-PHONE-NUMBER-ARR       TO  CL-DATA             00095000
        ELSE                                                       00095100
          MOVE  SPACES                     TO  CL-DATA             00095200
        END-IF                                                     00095300
        WRITE PRT-RECORD  FROM  CUST-LINE1 AFTER 1                 00095400
                                                                   00095500
       WHEN ORA-SQL-ROW-NOT-FOUND                                  00095600
         SET BRC-SUCCESSFUL-COMPLETION  TO TRUE                    00095700
                                                                   00095800
       WHEN OTHER                                                  00095900
         SET BRC-SQL-ERROR  TO  TRUE                               00096000
         DISPLAY                                                   00096100
             'I/O ERROR - CUSTOMER SELECT  '                       00096200
     END-EVALUATE                                                  00096300
     .                                                             00096400
 P530-PRINT-CUST-DATA-EXIT.                                        00096500
```

```
      EXIT.                                                              00096600
*                                                                        00096700
**************************************************************************00096800
*  PRINTS NO BENEFITS PROCESSED BANNER MESSAGE                           *00096900
**************************************************************************00097000
 P550-NO-PROC-MSG.                                                       00097100
                                                                         00097200
*    DISPLAY 'P550:  START'                                             00097300
                                                                         00097400
*    WRITE PRT-RECORD  FROM  BAN1-L1 AFTER 3                            00097500
*    WRITE PRT-RECORD  FROM  BAN1-L2 AFTER 1                            00097600
     WRITE PRT-RECORD  FROM  BAN1-L3 AFTER 3                            00097700
*    WRITE PRT-RECORD  FROM  BAN1-L2 AFTER 1                            00097800
*    WRITE PRT-RECORD  FROM  BAN1-L1 AFTER 1                            00097900
     .                                                                  00098000
 P550-NO-PROC-MSG-EXIT.                                                 00098100
      EXIT.                                                             00098200
/                                                                       00098300
**************************************************************************00098400
*  CHECKS FOR DATABASE ERRORS                                            *00098500
*  IF DATABASE ERROR DISPLAYS MESSAGE TO SYSOUT                          *00098600
**************************************************************************00098700
 Q000-CHECK-SQLCODE.                                                    00098800
                                                                         00098900
*    DISPLAY 'Q000:  START'                                             00099000
                                                                         00099100
     MOVE SQLCODE                      TO ORA-NAMED-SQLCODE             00099200
                                          ORA-SQLCODE-DISP-4            00099300
     EVALUATE TRUE                                                      00099400
       WHEN ORA-SQL-SUCCESSFUL                                          00099500
       WHEN ORA-SQL-ROW-NOT-FOUND                                       00099600
         CONTINUE                                                       00099700
                                                                        00099800
*                                                                       00099900
*           DATABASE ERROR                                              00100000
*                                                                       00100100
       WHEN  OTHER                                                      00100200
         SET  WS-DB-ERROR  TO  TRUE                                     00100300
         DISPLAY                                                        00100400
                ' *** SQL ERROR *** '                                   00100500
                ' SQLCODE :'      ORA-SQLCODE-DISP-4                    00100600
                ' MESSAGE :'      SQLERRMC                              00100700
                ' PARA: '         WORK-PARA                             00100800
                ' TABLE: '        ORA-TABLE-ID                          00100900
                ' FUNCTION: '     ORA-FUNCTION-ID                       00101000
     END-EVALUATE                                                       00101100
     .                                                                  00101200
 Q000-CHECK-SQLCODE-EXIT.                                               00101300
     EXIT.                                                              00101400
/                                                                       00101500
 V100-OPEN-BEN-LOAD.                                                    00101600
                                                                        00101700
*    DISPLAY 'V100:  START'                                             00101800
     MOVE    'V100:   '                TO  WORK-PARA                    00101900
     MOVE    'OPEN CUR'                TO  ORA-FUNCTION-ID              00102000
     MOVE    'BENEFIT-LOAD'            TO  ORA-TABLE-ID                 00102100
*                                                                       00102200
```

```
           EXEC SQL  OPEN                                                  00102300
              BENEFITS_LOAD                                               00102400
           END-EXEC                                                       00102500
                                                                          00102600
           PERFORM  Q000-CHECK-SQLCODE                                    00102700
              THRU  Q000-CHECK-SQLCODE-EXIT                               00102800
                                                                          00102900
           IF ORA-SQL-SUCCESSFUL                                          00103000
              CONTINUE                                                    00103100
           ELSE                                                           00103200
              SET BRC-SQL-ERROR TO TRUE                                   00103300
           END-IF                                                         00103400
           .                                                              00103500
       V100-OPEN-BEN-LOAD-EXIT.                                           00103600
           EXIT.                                                          00103700
      /                                                                   00103800
       V150-FETCH-BENEFITS-LOAD.                                          00103900
                                                                          00104000
      *    DISPLAY 'V150:  START'                                         00104100
           MOVE    'V150:   '                TO  WORK-PARA                00104200
           MOVE    'FETCHCUR'                TO  ORA-FUNCTION-ID          00104300
           MOVE    'BENEFIT-LOAD'            TO  ORA-TABLE-ID             00104400
      *                                                                   00104500
           EXEC SQL   FETCH   BENEFITS_LOAD                               00104600
             INTO                                                         00104700
-INC CWELB805                                                             0010480%
           END-EXEC                                                       00104900
                                                                          00105000
           PERFORM  Q000-CHECK-SQLCODE                                    00105100
              THRU  Q000-CHECK-SQLCODE-EXIT                               00105200
           .                                                              00105300
       V150-FETCH-BENEFITS-LOAD-EXIT.                                     00105400
           EXIT.                                                          00105500
      /                                                                   00105600
       V200-SELECT-CUSTOMER.                                              00105700
      *    DISPLAY 'V200:  START'                                         00105800
           MOVE    'V200:   '                TO  WORK-PARA                00105900
           MOVE    'SELECT  '                TO  ORA-FUNCTION-ID          00106000
           MOVE    'CUSTOMER    '            TO  ORA-TABLE-ID             00106100
      *                                                                   00106200
           INITIALIZE  CUSTOMER-ROW                                       00106300
      *                                                                   00106400
           EXEC SQL   SELECT                                              00106500
-INC CWELB820                                                             0010660%
              INTO                                                        00106700
-INC CWELB825                                                             0010680%
             FROM  MCHECK.CUSTOMER                                        00106900
             WHERE  CUSTOMER_ID  =  :BEN-CUSTOMER-ID                      00107000
           END-EXEC                                                       00107100
                                                                          00107200
           PERFORM  Q000-CHECK-SQLCODE                                    00107300
              THRU  Q000-CHECK-SQLCODE-EXIT                               00107400
           .                                                              00107500
       V200-SELECT-CUSTOMER-EXIT.                                         00107600
           EXIT.                                                          00107700
```

APPENDIX E

```
*****************************************************************
 IDENTIFICATION DIVISION.
*****************************************************************
 PROGRAM-ID.    CWUCB500.
 AUTHOR.        CUBIC/CTS.
 INSTALLATION.
 DATE-WRITTEN.  DECEMBER 1999.
 DATE-COMPILED.
*****************************************************************
*                                                               *
* PROGRAM NAME  : CLAIM CONFIRMATION PROCESS (EUB5)             *
* PROGRAM ID    : CWUCB500.                                     *
*                                                               *
* SYSTEM:  9121-490  MVS/ESA, CICS, COBOL II, VSAM, ORACLE      *
* PROJECT: 170-2719 ELECTRONIC BENEFITS DISTRIBUTION SYSTEM     *
*                   ON-LINE MONITORING AND CONTROL.             *
*                                                               *
* DESC:    THIS PROGRAM HANDLES THE MESSAGE (EUB5), WHICH       *
*          CONFIRMS THE BENEFITS WRITTEN TO A PATRAON'S         *
*          GO CARD.                                             *
*                                                               *
*          UPDATES EACH BENEFIT ROW THAT WAS IDENTIFIED IN THE  *
*          EUB5 MESSAGE.                                        *
*                                                               *
*          INSERT A CLAIMS ROW FOR EACH BENEFIT ROW UPDATED.    *
*                                                               *
*          LOG BEFORE/AFTER IMAGES OF EACH ROW INSERTED/UPDATED.*
*                                                               *
* INPUTS:  MESSAGE LINKAGE AREA (PASS-AREA)                     *
*          BENEFITS TABLE                (ORACLE)               *
*                                                               *
* OUTPUTS: MESSAGE LINKAGE AREA (PASS-AREA MACK)                *
*          CLAIMS TABLE                  (ORACLE)               *
*          BENEFITS TABLE                (ORACLE)               *
*          SYSTEM LOG                                           *
*                                                               *
* ERRORS:  PASSED TO CWAC5300 IN 'MLA-RETURN-CODE'              *
*          ALL ERRORS FROM THIS PROCESS WILL BE LOGGED IN THE   *
*          SYSTLOG FILES.                                       *
*                                                               *
*****************************************************************
* REVISION HISTORY :                                            *
*****************************************************************00004200
* 02/22/00 SMB000 RONO COMMENTED OUT ALL DEBUG LOGGER PERFORMS   *00004400
*****************************************************************00004200
* 12/15/99 SMB000 RONO INITIAL                                  *
*****************************************************************
/
 ENVIRONMENT DIVISION.
 DATA DIVISION.
 WORKING-STORAGE SECTION.

 01  WS-WORKAREA.
     05  WS-PROGRAM-ID              PIC  X(08)  VALUE 'CWUCB500'.
     05  FILLER                     PIC  X(13)  VALUE
         ' COMPILED ON:'.
     05  WS-WHEN-COMPILED           PIC  X(16).
```

```
       01  WS-TRACE-AREA.
           05  FILLER                      PIC  X(16)  VALUE
               '***TRACE AREA***'.
           05  WORK-PARA                   PIC  X(08).
       *   05  TRACE-CYCLE                 PIC  9(05)  VALUE ZEROES.
       *   05  FILLER                      PIC  X(02)  VALUE '**'.
       *   05  PARA-ID                     PIC  X(08)
       *       OCCURS 40   TIMES
       *       INDEXED BY PARA-NDX.

       01  WS-PROCESS-AREA.
           COPY CWWL0080.
      /
       01  WS-MISC-FIELDS.                                                  00007000
           05  IDXD                        PIC  S9(4)  COMP.                00007100
           05  IDXR                        PIC  S9(4)  COMP.                00007200
       *  MAXIMUM NUMBER OF REQUESTS ALLOWED ON IN AN EUB5 MESSAGE          00007400
           05  WS-VALID-MAX-REQ-NUM        PIC  S9(4)  COMP VALUE 15.       00007400
           05  WS-MAX-REQ-NUM              PIC  S9(4)  COMP.                00007300
SMB000     05  WS-BEN-FETCH-CNT            PIC  S9(4)  COMP.                00007300
SMB000     05  WS-CLM-FETCH-CNT            PIC  S9(4)  COMP.                00007300
           05  WS-SAVE-AUTH-CDE            PIC  X(15).                      00007300
           05  WS-SAVE-RETR-REF-NUM        PIC  X(12).                      00007300
           05  WS-AUTHORIZATION-CDE.                                        00007300
             10  WS-AC-RU-NUM              PIC  999.                        00007300
             10  WS-AC-DEV-NUM             PIC  99.                         00007300
             10  WS-AC-GMT                 PIC  9(10).                      00007300
           05  WS-REQ-TOT-AMT              PIC  S9(8)  COMP.                00007500
           05  WS-REQ-TOT-AMT-C3           PIC  S9(5)V99 COMP-3.            00007500
           05  WS-OVER-CLAIM-AMT           PIC  S9(8)  COMP.                00050600
           05  WS-OVER-CLAIM-AMT-C3        PIC  S9(5)V99 COMP-3.            00050600
           05  WS-DIFF-AMT-C3              PIC  S9(5)V99 COMP-3.            00050600
           05  WS-BEN-REM-VAL-AMT          PIC  S9(8)  COMP.                00007600
           05  WS-CLM-REM-VAL-AMT          PIC  S9(8)  COMP.                00007600
           05  WS-EUB5-REQ-VAL-AMT         PIC  S9(3)V99 COMP-3.            00007700
           05  WS-SAVE-INP-MSG-ID          PIC  S9(8)  COMP.                00008300
           05  WS-CLAIM-UPD-FLAG           PIC  9.                          00008300
             88  WS-SET-CLAIM-UPD-FLAG  VALUE ZERO.                         00008300
             88  WS-CLAIM-UPDATED       VALUE  1.                           00008300
           05  WS-UPDATE-BENEFITS-FLAG     PIC  X  VALUE 'N'.               00007800
             88  WS-START-BENEFITS-ALLOC        VALUE 'N'.                  00007900
             88  WS-BENEFITS-NOT-UPDATED        VALUE 'N'.                  00008000
             88  WS-BENEFITS-UPDATED            VALUE 'Y'.                  00008100
           05  WS-APPLY-CREDIT-FLAG        PIC  X  VALUE 'N'.               00007800
             88  WS-STANDARD-CLAIM              VALUE 'N'.                  00008000
             88  WS-APPLY-CREDIT-CLAIM          VALUE 'Y'.                  00008100
      /
       01  SYSTEM-LOGGER-COMMAREA.
           COPY CWLL3100.

       01  MESSAGE-ROUTER-LINKAGE.
           COPY CWLL3300.

       01  STOP-PROCESSING-COMMAREA.
           COPY CWLL5800.
```

```
       01  COMMON-DEFINITIONS.
           COPY CWWL0020.

       01  SK-CICS-REFORMAT-AREA.
           COPY CWWL9090.
      /
       01  EUB5-DATA.                                                  %
-INC CWULB500                                                         %
      /                                                            00011500
      *    SQL COPYBOOKS                                           00011600
                                                                   00011700
       01  ORACLE-SQL-CODES.                                       00011800
           COPY CWELB000.                                          00011900
                                                                   00012000
           EXEC SQL
               BEGIN DECLARE SECTION
           END-EXEC.

      *       BENEFITS HOST VARIABLES
-INC CWELB100                                                         %

      *       BENEFITS EQUIVALENCIES
-INC CWELB101                                                         %

      *       BENEFITS DEFINITION HOST VARIABLES
-INC CWELB130                                                         %

      *       BENEFITS DEFINITION EQUIVALENCIES
-INC CWELB131                                                         %

      *       CLAIMS  HOST VARIABLES
-INC CWELB500                                                         %

      *       CLAIMS  EQUIVALENCIES
-INC CWELB501                                                         %

       01  WS-SQL-HOST-VARIABLES.
         05  WS-PROCESS-DATE                 PIC   X(7).
         05  WS-PROCESS-DATE-X14             PIC   X(14).          00013500
         05  WS-PD-X14  REDEFINES WS-PROCESS-DATE-X14.             00013500
           10  WS-PD-DATE.                                         00013500
             15  WS-PD-CC                      PIC  S99.           00013500
             15  WS-PD-YY                      PIC  S99.           00013500
             15  WS-PD-MM                      PIC  S99.           00013500
             15  WS-PD-DD                      PIC  S99.           00013500
           10  WS-PD-TIME.                                         00013500
             15  WS-PD-HH                      PIC  S99.           00013500
             15  WS-PD-MI                      PIC  S99.           00013500
             15  WS-PD-SS                      PIC  S99.           00013500
         05  WS-ORA-SERIAL-NUM               PIC   X(15).          00014000
         05  WS-AUTH-CDE                     PIC   X(15).          00014000

      *  SQL DATA EQUIVALENCING AREA
           EXEC SQL VAR
               WS-PROCESS-DATE    IS  DATE
           END-EXEC
```

```
           EXEC SQL
               INCLUDE SQLCA
           END-EXEC

           EXEC SQL
               END DECLARE SECTION
           END-EXEC
      /
      *  CURSOR DECLARES
      * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      *  SELECT ALL BENEFITS FOR AUTHORIZATION CODE BEING CLAIMED
      * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
           EXEC SQL DECLARE   BENEFITS_AUTH    CURSOR FOR
             SELECT
-INC CWELB800                                                          %
              FROM  MCHECK.BENEFITS
               WHERE  SERIAL_NUM       =  :WS-ORA-SERIAL-NUM
                 AND  LAST_AUTH_CDE    =  :WS-AUTH-CDE
               ORDER BY  EXPIRATION_DATE
               FOR UPDATE OF                                        00017200
-INC CWELB800                                                          %
           END-EXEC

      * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      *  SELECT ALL CLAIMS FOR AUTHORIZATION CODE BEING CLAIMED
      * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
           EXEC SQL DECLARE   CLAIMS_AUTH       CURSOR FOR
             SELECT
-INC CWELB810                                                          %
              FROM  MCHECK.CLAIMS
               WHERE  SERIAL_NUM           =  :WS-ORA-SERIAL-NUM
                 AND  AUTHORIZATION_CDE    =  :WS-AUTH-CDE
               ORDER BY  EXPIRATION_DATE
           END-EXEC

      /
       LINKAGE SECTION.

       01  DFHCOMMAREA.
           03 MESSAGE-LINK-AREA.
           COPY CWLL5300.
      /
       PROCEDURE DIVISION.
      ****************************************************************
      *                      PROCEDURE DIVISION                      *
      ****************************************************************

       A000-MAIN-ROUTINE.

           MOVE 'A000' TO WORK-PARA.

           MOVE WHEN-COMPILED            TO WS-WHEN-COMPILED.
           INITIALIZE MESSAGE-ROUTER-LINKAGE.

           EXEC CICS HANDLE ABEND
                     LABEL (PROCABEC-ABEND-HANDLER)
```

```
     END-EXEC.

     MOVE +1                          TO SLC-MESSAGE-LENGTH.
     MOVE SPACES                      TO SLF-MESSAGE-VALUE.
     SET  MACK-BLANK                  TO TRUE.

*****************************************************************
*  MLA-RC-MISCELLANEOUS USED AS AN IN-PROCESS INDICATOR         *
*  IF NOT CHANGED BY THE PROGRAM, THE PROCESS WAS A SUCCESS     *
*****************************************************************
     IF  EIBCALEN > ZEROES
         SET  MLA-RC-MISCELLANEOUS TO TRUE
     ELSE
         SET  MLA-RC-NO-COMMAREA   TO TRUE
         STRING 'B501: '
                'PROGRAM: ' WS-PROGRAM-ID
                ' WAS NOT EXECUTED, NO COMMAREA WAS PASSED.'
                ' PARA: '            WORK-PARA
                DELIMITED BY SIZE
                     INTO SLF-MESSAGE-VALUE
            WITH POINTER SLC-MESSAGE-LENGTH
         PERFORM PROCABEC-LINK-TO-LOGGER
            THRU PROCABEC-LINK-TO-LOGGER-EXIT
         EXEC CICS RETURN END-EXEC
     END-IF

     SET  WS-PROCESS-START   TO  TRUE

     MOVE MLA-INPUT-DATA                TO  EUB5-DATA
     PERFORM E100-EDIT-EUB5-MSG THRU
             E100-EDIT-EUB5-MSG-EXIT

     IF WS-PROCESS-COMPLETE
       GO TO A100-END-PROGRAM
     END-IF

     EXEC SQL  SELECT SYSDATE
        INTO  :WS-PROCESS-DATE
      FROM  DUAL
     END-EXEC
                                                                  00030400
     EXEC SQL  SELECT TO_CHAR(SYSDATE, 'YYYYMMDDHH24MISS')        00030000
        INTO  :WS-PROCESS-DATE-X14                                00030100
      FROM  DUAL                                                  00030200
     END-EXEC                                                     00030300

     EVALUATE  TRUE
      WHEN  EUB5-COMPLETED-OK
       PERFORM B100-PROCESS-CLAIMS THRU
               B100-PROCESS-CLAIMS-EXIT

      WHEN  EUB5-VALUE-NOT-USED
      WHEN  EUB5-NOT-PROCESSED
      WHEN  EUB5-EV-FAILURE
       CONTINUE

      WHEN  OTHER
```

```
      SET  WS-EDIT-ERROR   TO  TRUE
      SET  MLA-RC-BAD-MLA-TRANS-ID  TO  TRUE
    END-EVALUATE
    .
A100-END-PROGRAM.

                                                                        00023800
    IF WS-DB-ERROR                                                      00023900
        MOVE  'BENEFIT/CLAIMS'              TO  ORA-TABLE-ID            00039100
        MOVE  'ROLLBACK'              TO  ORA-FUNCTION-ID               00039100
        MOVE +1                       TO SLC-MESSAGE-LENGTH
        MOVE SPACES                   TO SLF-MESSAGE-VALUE
        STRING 'B109:'
               ' ROLLBACK PROCESSED '
               ' SQLCODE :'      ORA-SQLCODE-DISP-4
               ' MESSAGE :'      SQLERRMC
               ' PARA: '         WORK-PARA
               ' BEN ROW: '      BENEFITS-ROW
               ' CLM ROW: '      CLAIMS-ROW
                 DELIMITED BY SIZE
                      INTO SLF-MESSAGE-VALUE
              WITH POINTER SLC-MESSAGE-LENGTH
          PERFORM PROCABEC-LINK-TO-LOGGER
             THRU PROCABEC-LINK-TO-LOGGER-EXIT
      PERFORM B530-ROLLBACK-TABLES THRU                                 00024000
              B530-ROLLBACK-TABLES-EXIT                                 00024000
    ELSE                                                                00023900
      IF  WS-PROCESS-SUCCESSFUL                                         00023900
        MOVE +1                       TO SLC-MESSAGE-LENGTH
        MOVE SPACES                   TO SLF-MESSAGE-VALUE
        STRING 'B110:'
               ' COMMIT   PROCESSED '
               ' BEN ROW: '      BENEFITS-ROW
               ' CLM ROW: '      CLAIMS-ROW
                 DELIMITED BY SIZE
                      INTO SLF-MESSAGE-VALUE
              WITH POINTER SLC-MESSAGE-LENGTH
*         PERFORM PROCABEC-LINK-TO-LOGGER
*            THRU PROCABEC-LINK-TO-LOGGER-EXIT
        PERFORM B540-COMMIT-DB-ACTIVITY THRU                            00056900
                B540-COMMIT-DB-ACTIVITY-EXIT                            00056900
      END-IF                                                            00023900
    END-IF                                                              00024200

    MOVE 'A100' TO WORK-PARA
                                                                        00024800
    EXEC SQL                                                            00024900
         CLOSE BENEFITS_AUTH                                            00024900
    END-EXEC                                                            00024900
                                                                        00024800
    EXEC SQL                                                            00024900
         CLOSE CLAIMS_AUTH                                              00024900
    END-EXEC                                                            00024900

    IF  MLA-RC-MISCELLANEOUS
     SET  MLA-RC-GOOD
          MACK-PROCESSED      TO  TRUE
```

```
     END-IF

*
*    TRANSFER CONTROL TO THE OUTPUT TRANSACTION MONITOR (OTM)
*
     SET CD-OTM                              TO TRUE.

     EXEC CICS XCTL
          PROGRAM  (CD-NAMED-PGM)
          COMMAREA (MESSAGE-LINK-AREA)
          LENGTH   (LENGTH OF MESSAGE-LINK-AREA)
          RESP     (WS-RESP)
     END-EXEC

     IF  WS-RESP NOT = DFHRESP (NORMAL)
         EXEC CICS DUMP
                   DUMPCODE (MLA-TRANS-ID)
                   COMPLETE
         END-EXEC
         EXEC CICS RETURN END-EXEC
     END-IF
     .
 A100-END-PROGRAM-EXIT.
     EXIT.
/
*****************************************************************
*  PROCESS EACH BENEFIT AGAINST THE CLAIMS CONTAINED IN THE EUB5 *
*  MESSAGE.                                                      *
*  INSERT A CLAIM ROW FOR THE AMOUNT THAT THE BENEFIT CONTRIBUTED*
*  TO THE CLAIM AMOUNT.                                          *
*****************************************************************
 B100-PROCESS-CLAIMS.

     MOVE 'BENEFITS'                  TO  ORA-TABLE-ID           00068100
     MOVE 'OPECUR AUTHORIZATION'      TO  ORA-FUNCTION-ID        00068100
     MOVE  'B100'                     TO  WORK-PARA
                                                                 00068000
     PERFORM  TRACE-IT                                           00068100

*    OPEN AUTHORIZATION CURSOR
     MOVE  EUB5-SERIAL-NUM            TO  WS-SERIAL-NUM
     MOVE  WS-SN-HI-2                 TO  WS-OSN-HI-5            00032800
     MOVE  WS-SN-LO-4                 TO  WS-OSN-LO-10           00032900
     MOVE  WS-SERIAL-NUM-X15          TO  WS-ORA-SERIAL-NUM      00032900
     MOVE  EUB5-AUTH-CDE              TO  WS-AUTH-CDE

     EXEC SQL
        OPEN   BENEFITS_AUTH
     END-EXEC

     PERFORM Q000-CHECK-SQLCODE THRU
             Q000-CHECK-SQLCODE-EXIT

     IF  WS-DB-ERROR
       SET MLA-RC-FAILED-DB-IO  TO  TRUE
       GO TO  B100-PROCESS-CLAIMS-EXIT
     END-IF
```

```
                                                                        00068000
           MOVE 'BENEFITS'                  TO  ORA-TABLE-ID            00068100
           MOVE 'FTCH   BENEFITS_AUTH'      TO  ORA-FUNCTION-ID         00068100
           PERFORM   UNTIL  WS-PROCESS-COMPLETE                         00068100
             MOVE 'B100FTCH'                TO  WORK-PARA               00068100
             PERFORM  TRACE-IT                                          00068100

             EXEC SQL  FETCH   BENEFITS_AUTH                            00034700
                INTO                                                    00034800
-INC CWELB805                                                           0003490%
             END-EXEC                                                   00035000
                                                                        00035100
             PERFORM  Q000-CHECK-SQLCODE THRU                           00035200
                      Q000-CHECK-SQLCODE-EXIT                           00035300
                                                                        00035400
             EVALUATE   TRUE                                            00035500
                                                                        00038900
              WHEN  ORA-SQL-SUCCESSFUL                                  00035600
                                                                        00036600
SMB000          ADD  1  TO  WS-BEN-FETCH-CNT                            00036700
                SET  BEN-AVAILABLE  TO  TRUE                            00036700
                MOVE     +0                TO  BEN-LAST-CLAIM-VAL-AMT   00036700
                MOVE  WS-PROCESS-DATE      TO  BEN-LAST-CLAIM-DT-TM     00036700
                                               BEN-LAST-REQUEST-DT-TM   00036700
                SET  BEN-CLAIM-CONFIRM  TO TRUE                         00036700
                PERFORM B380-SELECT-BEN-DEF THRU                        00036700
                        B380-SELECT-BEN-DEF-EXIT                        00036800
                IF  ORA-SQL-SUCCESSFUL                                  00036900
                  COMPUTE  WS-BEN-REM-VAL-AMT                           00041400
                                        =  BEN-REM-VAL-AMT  *  100      00041400
                  MOVE  ZERO               TO  WS-REQ-TOT-AMT           00037000
                      MOVE 'B130    '               TO  WORK-PARA       00068100
                      PERFORM  TRACE-IT                                 00068100
                  PERFORM  B130-GET-CLAIM-AMT THRU                      00037100
                           B130-GET-CLAIM-AMT-EXIT                      00037100
                    VARYING  IDXR  FROM  1  BY  1                       00037300
                        UNTIL  IDXR  >  WS-MAX-REQ-NUM                  00037400
                END-IF                                                  00036900
                                                                        00038900
                IF  WS-REQ-TOT-AMT  >  ZERO                             00008300
                  COMPUTE  WS-REQ-TOT-AMT-C3  =  WS-REQ-TOT-AMT / 100   00008300
                  SUBTRACT  WS-REQ-TOT-AMT-C3  FROM  BEN-REM-VAL-AMT    00068100
                  MOVE  WS-REQ-TOT-AMT-C3   TO  BEN-LAST-CLAIM-VAL-AMT  00068100
                  SET  BEN-AVAILABLE  TO  TRUE                          00068100

      *  MOVE  DATA FROM BEN-  TO  CLM-                                 00049800
                  MOVE  BEN-CUSTOMER-ID          TO  CLM-CUSTOMER-ID    00049900
                  MOVE  BEN-BENEFIT-TYPE         TO  CLM-BENEFIT-TYPE   00050000
                  MOVE  BEN-SERIAL-NUM           TO  CLM-SERIAL-NUM     00050100
                  MOVE  BEN-EFFECTIVE-DATE                              00050200
                                              TO  CLM-EFFECTIVE-DATE    00050200
                  MOVE  BEN-LAST-AUTH-CDE                               00050200
                                              TO  CLM-AUTHORIZATION-CDE 00050200
                  MOVE  BEN-MFG-SERIAL-NUM                              00050400
                                              TO  CLM-MFG-SERIAL-NUM    00050400
                  MOVE  BEN-EXPIRATION-DATE                             00050500
                                              TO  CLM-EXPIRATION-DATE   00050500
```

```
                MOVE  BEN-LAST-CLAIM-VAL-AMT     TO  CLM-CLAIM-VAL-AMT00050600
                MOVE  BEN-LAST-RETR-REF-NUM      TO  CLM-RETR-REF-NUM 00050700
                MOVE  WS-PROCESS-DATE            TO  CLM-REQUEST-DT-TM00036700
                SET  CLM-CLAIM-CONFIRM   TO TRUE                      00050800
                PERFORM B150-CREATE-CLAIM-ROW THRU                    00024000
                        B150-CREATE-CLAIM-ROW-EXIT                    00024000
              END-IF
                                                                      00038900
              IF  WS-IN-PROCESS
            MOVE 'BENEFITS'                  TO  ORA-TABLE-ID         00068100
            MOVE 'UPDATE AFTER CLMUPD*'      TO  ORA-FUNCTION-ID      00068100
                 MOVE 'B180    '                TO  WORK-PARA         00068100
                PERFORM  TRACE-IT                                     00068100
                SET  BEN-AVAILABLE  TO  TRUE                          00068100
                PERFORM B180-UPDATE-BENEFIT THRU                      00024000
                        B180-UPDATE-BENEFIT-EXIT                      00024000
              END-IF
                                                                      00038900
             WHEN  ORA-SQL-END-OF-FETCH                               00039000
                MOVE 'B100 EOF'                TO  WORK-PARA          00068100
                PERFORM  TRACE-IT                                     00068100
                                                                      00038900
SMB000        IF  WS-BEN-FETCH-CNT  >  ZERO                           00024000
                PERFORM B190-CREATE-OVER-CLAIM THRU                   00024000
                        B190-CREATE-OVER-CLAIM-EXIT                   00024000
SMB000*       IF  WS-IN-PROCESS                                       00039100
SMB000*         SET  WS-PROCESS-SUCCESSFUL  TO  TRUE                  00039100
SMB000*       END-IF                                                  00039100
SMB000        ELSE                                                    00024000
SMB000          PERFORM  B200-APPLY-CREDIT THRU                       00024000
SMB000                   B200-APPLY-CREDIT-EXIT                       00024000
SMB000        END-IF                                                  00024000
SMB000                                                                00024000
SMB000        IF  WS-IN-PROCESS                                       00039100
SMB000          SET  WS-PROCESS-SUCCESSFUL  TO  TRUE                  00039100
SMB000        END-IF                                                  00039100
                                                                      00038900
             WHEN  OTHER                                              00039000
                MOVE 'B100OTHR'                TO  WORK-PARA          00068100
                PERFORM  TRACE-IT                                     00068100
              SET  WS-DB-ERROR  TO  TRUE                              00039100
              SET  MLA-RC-FAILED-DB-IO  TO  TRUE                      00039100
            END-EVALUATE                                              00039500
          END-PERFORM                                                 00039500
          .
       B100-PROCESS-CLAIMS-EXIT.
           EXIT.

      ****************************************************************
      *  VALIDATE THE PRODUCT/PURSE FOR THIS BENEFIT.                *
      *  SUM CLAIM AGAINST THIS BENEFIT                              *
      ****************************************************************
       B130-GET-CLAIM-AMT.

           MOVE  'B130'                      TO  WORK-PARA
                                                                      00040800
           IF  EUB5-REQ-VAL-AMT (IDXR)  >  ZERO                       00040900
```

```
       MOVE  ZERO                          TO  WS-BIN-2N              00041000
       MOVE  EUB5-PURSE-CLASS-ID (IDXR) TO  WS-B2-LO                  00041100
       MOVE  WS-BIN-2N                     TO  IDXD                   00041200
       IF  BND-BENEFIT-ALLOWED (IDXD)                                 00041300
         COMPUTE  WS-EUB5-REQ-VAL-AMT                                 00041500
                          =  EUB5-REQ-VAL-AMT (IDXR)  /  100          00041600
                                                                      00041700
         EVALUATE  TRUE                                               00041800
          WHEN  WS-BEN-REM-VAL-AMT <= EUB5-REQ-VAL-AMT (IDXR)         00041900
            ADD  WS-BEN-REM-VAL-AMT TO  WS-REQ-TOT-AMT                00042000
*           ADD  WS-EUB5-REQ-VAL-AMT  TO  BEN-LAST-CLAIM-VAL-AMT      00042200
            SUBTRACT WS-BEN-REM-VAL-AMT                               00042300
                                FROM  EUB5-REQ-VAL-AMT (IDXR)         00042400
            MOVE  ZERO                  TO  WS-BEN-REM-VAL-AMT        00042500
                                                                      00042600
          WHEN  OTHER                                                 00042700
            ADD  EUB5-REQ-VAL-AMT (IDXR) TO  WS-REQ-TOT-AMT           00042800
*           ADD  WS-EUB5-REQ-VAL-AMT                                  00043100
*                                    TO  BEN-LAST-CLAIM-VAL-AMT       00043200
            SUBTRACT  WS-EUB5-REQ-VAL-AMT                             00043400
                                FROM   WS-BEN-REM-VAL-AMT             00043300
            MOVE  ZERO                  TO  EUB5-REQ-VAL-AMT (IDXR)   00043500
         END-EVALUATE                                                 00043600
       END-IF                                                         00043700
     END-IF                                                           00044300
     .
 B130-GET-CLAIM-AMT-EXIT.
     EXIT.

*****************************************************************
*  VALIDATE THE PRODUCT/PURSE FOR THIS BENEFIT.                 *
*  SUM CREDIT TO BE APPLIED TO THIS BENEFIT                     *
*****************************************************************
 B135-GET-CREDIT-AMT.

     MOVE  'B135'                        TO  WORK-PARA
                                                                      00040800
     MOVE  ZERO                          TO  WS-BIN-2N              00041000
     MOVE  EUB5-PURSE-CLASS-ID (IDXR) TO  WS-B2-LO                  00041100
     MOVE  WS-BIN-2N                     TO  IDXD                   00041200
     IF  BND-BENEFIT-ALLOWED (IDXD)                                 00041300
       COMPUTE  WS-EUB5-REQ-VAL-AMT                                 00041500
                        =  EUB5-REQ-VAL-AMT (IDXR)  /  100          00041600
                                                                      00041700
       EVALUATE  TRUE                                               00041800
        WHEN  WS-CLM-REM-VAL-AMT <= EUB5-REQ-VAL-AMT (IDXR)         00041900
          ADD  WS-CLM-REM-VAL-AMT TO  WS-REQ-TOT-AMT                00042000
          SUBTRACT WS-CLM-REM-VAL-AMT                               00042300
                              FROM  EUB5-REQ-VAL-AMT (IDXR)         00042400
          MOVE  ZERO                  TO  WS-CLM-REM-VAL-AMT        00042500
                                                                      00042600
        WHEN  OTHER                                                 00042700
          ADD  EUB5-REQ-VAL-AMT (IDXR) TO  WS-REQ-TOT-AMT           00042800
          SUBTRACT  WS-EUB5-REQ-VAL-AMT                             00043400
                              FROM   WS-CLM-REM-VAL-AMT             00043300
          MOVE  ZERO                TO  EUB5-REQ-VAL-AMT (IDXR)     00043500
       END-EVALUATE                                                 00043600
```

```
           END-IF                                                        00043700
           .
       B135-GET-CREDIT-AMT-EXIT.
           EXIT.

      ******************************************************************
      *  CREATE A CLAIM ROW FOR THE BENEFITS USFED                     *
      ******************************************************************
       B150-CREATE-CLAIM-ROW.

           MOVE  'B150'                      TO  WORK-PARA
           MOVE 'CLAIMS'                     TO  ORA-TABLE-ID            00068100
           MOVE 'INSERT'                     TO  ORA-FUNCTION-ID         00068100
           PERFORM  TRACE-IT                                             00068100
                                                                         00051000
      *  THEN INSERT THE CLAIM ROW INTO THE CLAIM TABLE                  00051100
           EXEC SQL INSERT                                               00051200
             INTO  MCHECK.CLAIMS                                         00051300
                   (                                                     0005140%
-INC CWELB810                                                            0005140%
                   )                                                     00051800
             VALUES                                                      00051600
                   (                                                     0005140%
-INC CWELB815                                                            0005170%
                   )                                                     00051800
           END-EXEC                                                      00051900
                                                                         00052000
           PERFORM  Q000-CHECK-SQLCODE THRU                              00052100
                    Q000-CHECK-SQLCODE-EXIT                              00052200
           .
       B150-CREATE-CLAIM-ROW-EXIT.
           EXIT.

      ******************************************************************
      *  UPDATE THE BENEFIT ROW BEING PROCESSED                        *
      ******************************************************************
       B180-UPDATE-BENEFIT.

           MOVE  'B180'                      TO  WORK-PARA

           EXEC  SQL  UPDATE  MCHECK.BENEFITS                            00036600
             SET                                                         00036600
-INC CWELB808                                                            0003660%
             WHERE  SERIAL_NUM       =   :BEN-SERIAL-NUM                 00036600
               AND  CUSTOMER_ID      =   :BEN-CUSTOMER-ID                00036600
               AND  BENEFIT_TYPE     =   :BEN-BENEFIT-TYPE               00036600
               AND  EFFECTIVE_DATE   =   :BEN-EFFECTIVE-DATE             00036600
           END-EXEC                                                      00036600
                                                                         00038900
           PERFORM  Q000-CHECK-SQLCODE THRU                              00035200
                    Q000-CHECK-SQLCODE-EXIT                              00035300
                                                                         00038900
           IF    NOT  ORA-SQL-SUCCESSFUL                                 00035600
             SET  WS-DB-ERROR  TO  TRUE                                  00039100
             MOVE  'BENEFITS'              TO  ORA-TABLE-ID              00039100
             MOVE  'UPDATE'                TO  ORA-FUNCTION-ID           00039100
             MOVE +1                       TO SLC-MESSAGE-LENGTH
```

```
       MOVE SPACES                  TO SLF-MESSAGE-VALUE
       STRING 'B506:'
              ' EUB5 NOT PROCESSED - DB ERROR'
              ' SQLCODE :'      ORA-SQLCODE-DISP-4
              ' MESSAGE :'      SQLERRMC
              ' PARA: '         WORK-PARA
              ' TABLE: '        ORA-TABLE-ID                    00075300
              ' FUNCTION: '     ORA-FUNCTION-ID                 00075300
              ' EUB5 DATA: '    EUB5-DATA
              ' BENE ROW : '    BENEFITS-ROW
                DELIMITED BY SIZE
                     INTO SLF-MESSAGE-VALUE
            WITH POINTER SLC-MESSAGE-LENGTH
         PERFORM PROCABEC-LINK-TO-LOGGER
            THRU PROCABEC-LINK-TO-LOGGER-EXIT
     END-IF                                                     00035500
     .
 B180-UPDATE-BENEFIT-EXIT.
     EXIT.

*****************************************************************
*  CREATE A CLAIM FOR EUB5 CLAIM AMOUNTS THAT ARE GREATER THAN   *
*  THE VALUE AVAILABLE ON THE BENEFIT.                           *
*****************************************************************
 B190-CREATE-OVER-CLAIM.

     MOVE  'B190'                    TO  WORK-PARA

     MOVE  ZERO                      TO  WS-OVER-CLAIM-AMT
     PERFORM  VARYING IDXR FROM  1  BY  1
                UNTIL IDXR  >  WS-MAX-REQ-NUM
       IF  EUB5-REQ-VAL-AMT (IDXR)  >    ZERO
         ADD  EUB5-REQ-VAL-AMT (IDXR)  TO  WS-OVER-CLAIM-AMT
       END-IF
     END-PERFORM

     IF  WS-OVER-CLAIM-AMT  >  ZERO

*  MOVE  DATA   TO  CLAIMS                                       00049800
       MOVE  BEN-CUSTOMER-ID         TO  CLM-CUSTOMER-ID        00049900
       MOVE  BEN-BENEFIT-TYPE        TO  CLM-BENEFIT-TYPE       00050000
       MOVE  BEN-SERIAL-NUM          TO  CLM-SERIAL-NUM         00050100
       MOVE  BEN-EFFECTIVE-DATE                                 00050200
                                     TO  CLM-EFFECTIVE-DATE     00050200
       MOVE  BEN-LAST-AUTH-CDE                                  00050200
                                     TO  WS-AUTHORIZATION-CDE   00050200
       ADD  1  TO  WS-AC-GMT                                    00050200
       MOVE  WS-AUTHORIZATION-CDE                               00050200
                                     TO  CLM-AUTHORIZATION-CDE  00050200
       MOVE  BEN-MFG-SERIAL-NUM                                 00050400
                                     TO  CLM-MFG-SERIAL-NUM     00050400
       MOVE  BEN-EXPIRATION-DATE                                00050500
                                     TO  CLM-EXPIRATION-DATE    00050500
       COMPUTE  WS-OVER-CLAIM-AMT-C3                            00050600
                              =  (WS-OVER-CLAIM-AMT  /  100)    00050500
       MOVE  WS-OVER-CLAIM-AMT-C3    TO  CLM-CLAIM-VAL-AMT      00050600
       MOVE  BEN-LAST-RETR-REF-NUM   TO  CLM-RETR-REF-NUM       00050700
```

```
           SET  CLM-CLAIM-CONFIRM  TO  TRUE                              00050800
           MOVE  WS-PROCESS-DATE          TO  CLM-REQUEST-DT-TM          00050900

           PERFORM B150-CREATE-CLAIM-ROW THRU                            00024000
                   B150-CREATE-CLAIM-ROW-EXIT                            00024000
         END-IF                                                          00024000
         .
       B190-CREATE-OVER-CLAIM-EXIT.
         EXIT.

      ****************************************************************00044800
      *  APPLIES A CREDIT TO THE BENEFIT - IF NEEDED                  *00044900
      ****************************************************************00045000
       B200-APPLY-CREDIT.                                              00045100
                                                                       00045200
         MOVE  'B200   *'                TO  WORK-PARA                 00075300
                                                                       00045200
         MOVE 'CLAIMS  '                 TO  ORA-TABLE-ID              00068100
         MOVE 'OPECUR CLAIMS_AUTH  '     TO  ORA-FUNCTION-ID           00068100
         PERFORM  TRACE-IT                                             00068100

         SET  WS-APPLY-CREDIT-CLAIM  TO  TRUE                          00008100

      *   OPEN CLAIMS_AUTH   CURSOR
         MOVE  EUB5-SERIAL-NUM           TO  WS-SERIAL-NUM
         MOVE  WS-SN-HI-2                TO  WS-OSN-HI-5               00032800
         MOVE  WS-SN-LO-4                TO  WS-OSN-LO-10              00032900
         MOVE  WS-SERIAL-NUM-X15         TO  WS-ORA-SERIAL-NUM         00032900
         MOVE  EUB5-AUTH-CDE             TO  WS-AUTH-CDE

         EXEC SQL
            OPEN   CLAIMS_AUTH
         END-EXEC

         PERFORM Q000-CHECK-SQLCODE THRU
                 Q000-CHECK-SQLCODE-EXIT

         IF  WS-DB-ERROR
           SET MLA-RC-FAILED-DB-IO  TO  TRUE
           GO TO  B100-PROCESS-CLAIMS-EXIT
         END-IF
                                                                       00068000
         MOVE 'CLAIMS  '                 TO  ORA-TABLE-ID              00068100
         MOVE 'FTCH   CLAIMS_AUTH  '     TO  ORA-FUNCTION-ID           00068100
         PERFORM   UNTIL  WS-PROCESS-COMPLETE                          00068100
           MOVE 'B200FTCH'               TO  WORK-PARA                 00068100
           PERFORM  TRACE-IT                                           00068100

           EXEC SQL  FETCH   CLAIMS_AUTH                               00034700
              INTO                                                     00034800
-INC CWELB815                                                          0003490%
           END-EXEC                                                    00035000
                                                                       00035100
           PERFORM  Q000-CHECK-SQLCODE THRU                            00035200
                    Q000-CHECK-SQLCODE-EXIT                            00035300
                                                                       00035400
           EVALUATE   TRUE                                             00035500
```

```
                                                                        00038900
             WHEN  ORA-SQL-SUCCESSFUL                                   00035600
                                                                        00068000
           MOVE 'CLAIMS  '                    TO  ORA-TABLE-ID         00068100
           MOVE 'FTCH   CLAIMS_AUTH  '        TO  ORA-FUNCTION-ID      00068100
             MOVE 'B200FCLM'                  TO  WORK-PARA            00068100
             PERFORM  TRACE-IT                                          00068100
                                                                        00036600
               ADD  1  TO  WS-CLM-FETCH-CNT                             00036700
               MOVE  CLM-CUSTOMER-ID        TO  BEN-CUSTOMER-ID         00036700
               MOVE  CLM-BENEFIT-TYPE       TO  BEN-BENEFIT-TYPE        00036700
               MOVE  CLM-SERIAL-NUM         TO  BEN-SERIAL-NUM          00036700
               MOVE  CLM-EFFECTIVE-DATE     TO  BEN-EFFECTIVE-DATE      00036700
               MOVE  CLM-AUTHORIZATION-CDE  TO  WS-SAVE-AUTH-CDE        00007300
               MOVE  CLM-RETR-REF-NUM       TO  WS-SAVE-RETR-REF-NUM    00050700
                                                                        00036600
               EXEC SQL SELECT
-INC CWELB800                                                           %
                 INTO
-INC CWELB805                                                           %
                 FROM  MCHECK.BENEFITS
                  WHERE  SERIAL_NUM       =  :BEN-SERIAL-NUM
                    AND  CUSTOMER_ID      =  :BEN-CUSTOMER-ID
                    AND  BENEFIT_TYPE     =  :BEN-BENEFIT-TYPE
                    AND  EFFECTIVE_DATE   =  :BEN-EFFECTIVE-DATE
                 FOR UPDATE OF                                          00017200
-INC CWELB800                                                           %
               END-EXEC
                                                                        00036600
               PERFORM  Q000-CHECK-SQLCODE THRU                         00035200
                        Q000-CHECK-SQLCODE-EXIT                         00035300
                                                                        00035400
               EVALUATE   TRUE                                          00035500
                                                                        00038900
                WHEN  ORA-SQL-SUCCESSFUL                                00035600
                  PERFORM  B250-UPD-BEN-INS-CLM THRU                    00035600
                           B250-UPD-BEN-INS-CLM-EXIT                    00035600
                                                                        00038900
                WHEN  OTHER                                             00035600
                  SET MLA-RC-FAILED-DB-IO  TO  TRUE
                  SET WS-DB-ERROR  TO  TRUE
      *                                                                 00074400
                  MOVE +1                        TO SLC-MESSAGE-LENGTH  00074700
                  MOVE SPACES                    TO SLF-MESSAGE-VALUE   00074800
                  STRING 'B507:'                                        00074900
                         ' SQLCODE :'      ORA-SQLCODE-DISP-4           00075000
                         ' PARA: '         WORK-PARA                    00075100
                         ' MESSAGE :'      SQLERRMC                     00075200
                         ' TABLE: '        ORA-TABLE-ID                 00075300
                         ' FUNCTION: '     ORA-FUNCTION-ID              00075300
                        DELIMITED BY SIZE                               00075400
                             INTO SLF-MESSAGE-VALUE                     00075500
                      WITH POINTER SLC-MESSAGE-LENGTH                   00075600
                  PERFORM PROCABEC-LINK-TO-LOGGER                       00075700
                     THRU PROCABEC-LINK-TO-LOGGER-EXIT                  00075800
                                                                        00036600
               END-EVALUATE                                             00035500
```

```
                                                                        00038900
        WHEN  ORA-SQL-END-OF-FETCH                                      00035600
          SET  WS-PROCESS-SUCCESSFUL  TO  TRUE
                                                                        00068000
    MOVE 'CLAIMS  '                   TO  ORA-TABLE-ID                00068100
    MOVE 'EOF    CLAIMS_AUTH  '       TO  ORA-FUNCTION-ID             00068100
      MOVE 'B200 EOF'                 TO  WORK-PARA                   00068100
      PERFORM  TRACE-IT                                                 00068100
                                                                        00038900
        WHEN  OTHER                                                     00035600
          SET MLA-RC-FAILED-DB-IO  TO  TRUE
          SET WS-DB-ERROR  TO  TRUE
                                                                        00038900
      END-EVALUATE                                                      00035500
                                                                        00036600
    END-PERFORM                                                         00068100
    .
 B200-APPLY-CREDIT-EXIT.                                                00045100
    EXIT.

*****************************************************************00044800
*  APPLIES CREDIT TO BENEFIT AMOUNT REMAINING                    *00044900
*  INSERTS A CLAIM FOR THE INCREASE ON THE BENEFITS ROW          *00044900
*****************************************************************00045000
 B250-UPD-BEN-INS-CLM.                                                  00045100
                                                                        00045200
    MOVE  'B250   *'                  TO  WORK-PARA                   00075300
                                                                        00045200
    PERFORM B380-SELECT-BEN-DEF THRU                                    00036700
            B380-SELECT-BEN-DEF-EXIT                                    00036800
                                                                        00045200
    IF  ORA-SQL-SUCCESSFUL                                              00036900
      SET  WS-APPLY-CREDIT-CLAIM  TO  TRUE                              00008100
      COMPUTE  WS-CLM-REM-VAL-AMT                                       00041400
                              =  CLM-CLAIM-VAL-AMT  *  100              00041400
      MOVE  ZERO                TO  WS-REQ-TOT-AMT                    00037000
               MOVE 'B250    '                TO  WORK-PARA            00068100
               PERFORM  TRACE-IT                                        00068100
      PERFORM  B135-GET-CREDIT-AMT THRU                                 00037100
               B135-GET-CREDIT-AMT-EXIT                                 00037100
        VARYING  IDXR  FROM  1  BY  1                                   00037300
            UNTIL  IDXR  >  WS-MAX-REQ-NUM                              00037400
                                                                        00038900
      IF  WS-REQ-TOT-AMT  >  ZERO                                       00008300
        COMPUTE  WS-REQ-TOT-AMT-C3  =  WS-REQ-TOT-AMT / 100             00008300
        COMPUTE  WS-DIFF-AMT-C3  =  CLM-CLAIM-VAL-AMT                   00068100
                                  -  WS-REQ-TOT-AMT-C3                  00068100
        IF  WS-DIFF-AMT-C3  NOT =   ZERO                                00068100
          COMPUTE  BEN-REM-VAL-AMT  =  BEN-REM-VAL-AMT                  00068100
                                    +  WS-DIFF-AMT-C3                   00068100

*   MOVE  DATA FROM BEN-  TO  CLM-                                      00049800
          MOVE  BEN-CUSTOMER-ID          TO  CLM-CUSTOMER-ID          00049900
          MOVE  BEN-BENEFIT-TYPE         TO  CLM-BENEFIT-TYPE         00050000
          MOVE  BEN-SERIAL-NUM           TO  CLM-SERIAL-NUM           00050100
          MOVE  BEN-EFFECTIVE-DATE                                      00050200
                                    TO  CLM-EFFECTIVE-DATE              00050200
```

```
                MOVE  WS-SAVE-AUTH-CDE                                  00050200
                                          TO  WS-AUTHORIZATION-CDE      00050200
                ADD  1  TO  WS-AC-GMT                                   00050200
                MOVE  WS-AUTHORIZATION-CDE                              00050200
                                          TO  CLM-AUTHORIZATION-CDE     00050200
                MOVE  BEN-MFG-SERIAL-NUM                                00050400
                                          TO  CLM-MFG-SERIAL-NUM        00050400
                MOVE  BEN-EXPIRATION-DATE                               00050500
                                          TO  CLM-EXPIRATION-DATE       00050500
                MOVE  ZERO                TO  CLM-CLAIM-VAL-AMT         00068100
                SUBTRACT  WS-DIFF-AMT-C3 FROM   CLM-CLAIM-VAL-AMT       00068100
                MOVE  WS-PROCESS-DATE     TO  CLM-REQUEST-DT-TM         00036700
                MOVE  WS-SAVE-RETR-REF-NUM  TO  CLM-RETR-REF-NUM        00050700
                SET  CLM-CLAIM-CREDIT   TO TRUE                         00050800
                PERFORM B150-CREATE-CLAIM-ROW THRU                      00024000
                        B150-CREATE-CLAIM-ROW-EXIT                      00024000
              END-IF
                                                                        00038900
              IF  WS-IN-PROCESS
            MOVE 'BENEFITS'                 TO  ORA-TABLE-ID            00068100
            MOVE 'UPD AFT CLM  CREDIT*'     TO  ORA-FUNCTION-ID         00068100
                   MOVE 'B180    '                TO  WORK-PARA         00068100
                 PERFORM  TRACE-IT                                      00068100
                IF  BEN-AVAILABLE                                       00068100
                  MOVE  WS-REQ-TOT-AMT-C3                               00068100
                                        TO  BEN-LAST-CLAIM-VAL-AMT      00068100
                  MOVE  WS-PROCESS-DATE   TO  BEN-LAST-CLAIM-DT-TM      00036700
                                              BEN-LAST-REQUEST-DT-TM    00036700
                  MOVE  WS-SAVE-RETR-REF-NUM                            00050700
                                       TO  BEN-LAST-RETR-REF-NUM        00050700
                  SET  BEN-CLAIM-CREDIT   TO TRUE                       00050800
                END-IF                                                  00068100
                                                                        00068100
                PERFORM B180-UPDATE-BENEFIT THRU                        00024000
                        B180-UPDATE-BENEFIT-EXIT                        00024000
              END-IF
            END-IF
            ELSE
              SET MLA-RC-FAILED-DB-IO  TO  TRUE
              SET WS-DB-ERROR  TO  TRUE
            END-IF
            .
         B250-UPD-BEN-INS-CLM-EXIT.                                     00045100
            EXIT.

        ***************************************************************00044800
        *  SELECTS ROW FROM BENEFIT DEFINITION TABLES                  *00044900
        ***************************************************************00045000
         B380-SELECT-BEN-DEF.                                           00045100
                                                                        00045200
            MOVE  'BEN DEF*'                TO  ORA-TABLE-ID            00075300
            MOVE  'SELECT'                  TO  ORA-FUNCTION-ID         00075300
                                                                        00045400
            EXEC SQL  SELECT                                            00045500
-INC CWELB830                                                           0004560%
              INTO                                                      00045700
-INC CWELB835                                                           0004580%
```

```
       FROM  MCHECK.BENEFITS_DEFINITION                                 00045900
        WHERE  BENEFIT_TYPE  =  :BEN-BENEFIT-TYPE                       00046000
     END-EXEC                                                           00046100
                                                                        00046200
     PERFORM  Q000-CHECK-SQLCODE THRU                                   00046300
              Q000-CHECK-SQLCODE-EXIT                                   00046400
*                                                                       00074400
     IF  NOT  ORA-SQL-SUCCESSFUL                                        00074500
         MOVE +1                      TO SLC-MESSAGE-LENGTH             00074700
         MOVE SPACES                  TO SLF-MESSAGE-VALUE              00074800
         STRING 'B507:'                                                 00074900
                ' SQLCODE :'       ORA-SQLCODE-DISP-4                   00075000
                ' MESSAGE :'       SQLERRMC                             00075200
                ' TABLE: '         ORA-TABLE-ID                         00075300
                ' FUNCTION: '      ORA-FUNCTION-ID                      00075300
                ' PARA: '          WORK-PARA                            00075100
               DELIMITED BY SIZE                                        00075400
                    INTO SLF-MESSAGE-VALUE                              00075500
            WITH POINTER SLC-MESSAGE-LENGTH                             00075600
         PERFORM PROCABEC-LINK-TO-LOGGER                                00075700
            THRU PROCABEC-LINK-TO-LOGGER-EXIT                           00075800
     END-IF                                                             00074500
     MOVE  'B380'                      TO  WORK-PARA                    00045300
     .                                                                  00046500
 B380-SELECT-BEN-DEF-EXIT.                                              00046600
     EXIT.                                                              00046700
                                                                        00056400
*****************************************************************     00056500
*  CLEARS EUB2 REQUEST AMOUNTS                                  *     00056600
*  PERFORMS ROLLBACK OF ALL DATABASE ACTIVITY                   *     00056700
*****************************************************************     00056800
 B530-ROLLBACK-TABLES.                                                  00056900
                                                                        00057000
     MOVE  'B530'                      TO  WORK-PARA                    00057100
                                                                        00057200
*  ROLLBACK THE DATABASE PROCESSING                                     00056500
     EXEC  SQL ROLLBACK                                                 00037300
         WORK                                                           00037400
     END-EXEC                                                           00037300
                                                                        00033300
     PERFORM  Q000-CHECK-SQLCODE THRU                                   00033400
              Q000-CHECK-SQLCODE-EXIT                                   00033500
     .                                                                  00056100
 B530-ROLLBACK-TABLES-EXIT.                                             00056900
     EXIT.                                                              00056300
                                                                        00056400
*****************************************************************     00056500
*  PERFORMS COMMIT   OF ALL DATABASE ACTIVITY                   *     00056700
*****************************************************************     00056800
 B540-COMMIT-DB-ACTIVITY.                                               00056900
                                                                        00057000
     MOVE  'B540'                      TO  WORK-PARA                    00057100
                                                                        00057200
*  ROLLBACK THE DATABASE PROCESSING                                     00056500
     EXEC  SQL COMMIT                                                   00037300
         WORK                                                           00037400
     END-EXEC                                                           00037300
```

```
                                                                        00033300
     PERFORM  Q000-CHECK-SQLCODE THRU                                   00033400
              Q000-CHECK-SQLCODE-EXIT                                   00033500
     .                                                                  00056100
 B540-COMMIT-DB-ACTIVITY-EXIT.                                          00056900
     EXIT.                                                              00056300
/
*****************************************************************
*  EDITS FOR:                                                   *
*  1. VALID MESSAGE ID                                          *
*****************************************************************
 E100-EDIT-EUB5-MSG.

     MOVE  '********'                 TO  ORA-TABLE-ID             00075300
     MOVE  'EDIT EUB5'                TO  ORA-FUNCTION-ID          00075300
     MOVE  'E100'                     TO  WORK-PARA
     PERFORM  TRACE-IT
*****************************************************************00067600
*  IS THIS AN EUB5 MESSAGE ????                                 *00067700
*****************************************************************00067800
                                                                        00067900
     IF  MLA-INPUT-TRANS-ID  NOT =  'EUB5'                              00068000
       SET  MLA-RC-BAD-MLA-TRANS-ID   TO  TRUE                          00068000
       MOVE +1                      TO SLC-MESSAGE-LENGTH               00068100
       MOVE SPACES                  TO SLF-MESSAGE-VALUE                00068200
       STRING 'B505: '                                                  00068300
              MLA-INPUT-TRANS-ID '<= INVALID TRANS ID! '                00068500
              ' MESSAGE DATA: '       EUB5-DATA                         00068600
              DELIMITED BY SIZE                                         00068700
                   INTO SLF-MESSAGE-VALUE                               00068800
           WITH POINTER SLC-MESSAGE-LENGTH                              00068900
       PERFORM PROCABEC-LINK-TO-LOGGER                                  00069000
          THRU PROCABEC-LINK-TO-LOGGER-EXIT                             00069100
       SET  WS-EDIT-ERROR  TO  TRUE                                     00069200
       GO TO  E100-EDIT-EUB5-MSG-EXIT                                   00069300
     END-IF                                                             00069400
                                                                        00069500
*****************************************************************00069600
*  DETERMINE NUMBER OF BENEFIT REQUESTS CONTAINED IN THE EUB5    *00069700
*  MESSAGE.                                                     *00069800
*****************************************************************00069900
     COMPUTE  WS-MAX-REQ-NUM  =                                         00070000
          ((MLA-INPUT-DATA-LENGTH  -  69) /  5)                         00070100
                                                                        00070400
     IF  WS-MAX-REQ-NUM  >  WS-VALID-MAX-REQ-NUM                        00070500
       SET  MLA-RC-BAD-MSG-LEN  TO  TRUE                                00068000
       MOVE  WS-MAX-REQ-NUM         TO WS-COMP-DISP-4                   00070600
       MOVE  WS-VALID-MAX-REQ-NUM   TO WS-COMP-DISP-4A                  00070700
       MOVE +1                      TO SLC-MESSAGE-LENGTH               00070800
       MOVE SPACES                  TO SLF-MESSAGE-VALUE                00070900
       STRING 'B504: '                                                  00071000
              ' MAX REQS EXCEEDED-'                                     00071100
              ' REQUEST COUNT: '           WS-COMP-DISP-4               00071200
              ' EXCEEDS MAXIMUM ALLOWED: ' WS-COMP-DISP-4A              00071300
              ' EUB5: '          EUB5-DATA                              00071400
              DELIMITED BY SIZE                                         00071500
                   INTO SLF-MESSAGE-VALUE                               00071600
```

```
           WITH POINTER SLC-MESSAGE-LENGTH                          00071700
         PERFORM PROCABEC-LINK-TO-LOGGER                            00071800
            THRU PROCABEC-LINK-TO-LOGGER-EXIT                       00071900
         SET  WS-EDIT-ERROR  TO  TRUE                               00072000
         GO TO  E100-EDIT-EUB5-MSG-EXIT                             00072100
       END-IF                                                       00072200
       .
 E100-EDIT-EUB5-MSG-EXIT.
       EXIT.

*****************************************************************
*  CHECKS FOR DATABASE ERRORS                                   *
*  IF DATABASE ERROR                                            *
*     LOGS THE ERROR                                            *
*****************************************************************
 Q000-CHECK-SQLCODE.

       MOVE SQLCODE                        TO ORA-NAMED-SQLCODE
                                              ORA-SQLCODE-DISP-4
       EVALUATE TRUE
         WHEN ORA-SQL-SUCCESSFUL
         WHEN ORA-SQL-ROW-NOT-FOUND
           CONTINUE

*
*            DATABASE ERROR
*
         WHEN  OTHER
           SET  WS-DB-ERROR  TO  TRUE
           MOVE +1                       TO SLC-MESSAGE-LENGTH
           MOVE SPACES                   TO SLF-MESSAGE-VALUE
           STRING 'B502:'
                  ' SQLCODE :'       ORA-SQLCODE-DISP-4
                  ' PARA: '          WORK-PARA
                  ' MESSAGE :'       SQLERRMC
                  ' PROGRAM '  WS-PROGRAM-ID
                 DELIMITED BY SIZE
                      INTO SLF-MESSAGE-VALUE
              WITH POINTER SLC-MESSAGE-LENGTH
           PERFORM PROCABEC-LINK-TO-LOGGER
              THRU PROCABEC-LINK-TO-LOGGER-EXIT
       END-EVALUATE
       .
 Q000-CHECK-SQLCODE-EXIT.
       EXIT.
/
 TRACE-IT.
         MOVE +1                        TO SLC-MESSAGE-LENGTH       00068100
         MOVE SPACES                    TO SLF-MESSAGE-VALUE        00068200
         STRING                                                     00068300
                ' WORK PARA: '        WORK-PARA                     00068600
                ' BENEFITS : '        BENEFITS-ROW                  00068600
                ' CLAIMS: '           CLAIMS-ROW                    00068600
                ' TABLE: '            ORA-TABLE-ID                  00068600
                ' FUCNTION: '         ORA-FUNCTION-ID               00068600
               DELIMITED BY SIZE                                    00068700
                    INTO SLF-MESSAGE-VALUE                          00068800
```

```
                WITH POINTER SLC-MESSAGE-LENGTH                          00068900
      *       PERFORM PROCABEC-LINK-TO-LOGGER                            00069000
      *          THRU PROCABEC-LINK-TO-LOGGER-EXIT                       00069100

      *     IF  PARA-NDX > 999
      *         ADD 1           TO TRACE-CYCLE
      *         SET PARA-NDX    TO 1
      *     ELSE
      *         SET PARA-NDX UP BY 1
      *     END-IF.
      *
      *     MOVE WORK-PARA      TO PARA-ID (1)
            .
       TRACE-IT-EXIT.
            EXIT.
      /
-INC CWPL9090                                                            %
      /
-INC CWXL9920                                                            %
      /
-INC CWXL9900                                                            %
```

APPENDIX F

```
*****************************************************************
 IDENTIFICATION DIVISION.
*****************************************************************
 PROGRAM-ID.    CWUCB200.
 AUTHOR.        CUBIC-CTS.
 INSTALLATION.
 DATE-WRITTEN.  NOVEMBER, 1999.
 DATE-COMPILED.
*****************************************************************
*                                                               *
* PROGRAM NAME  : BENEFITS RESPONSE (EUB2)                      *
* PROGRAM ID    : CWUCB200.                                     *
*                                                               *
* SYSTEM:  9121-490, MVS/ESA, CICS, COBOL II, VSAM.             *
* PROJECT: 170-2719, ELECTRONIC BENEFITS DISTRIBUTION SYSTEM    *
*                    ON-LINE MONITORING CONTROL.                *
*                                                               *
*                                                               *
* DESC:    THIS PROGRAM WILL BE USED TO TRANSMIT AN EUB2         *
*          TRANSACTION TO REMOTE UNITS.                         *
*          AN EUB2 TRANSACTION IS IN RESPONSE TO AN EUB1 BENEFITS*
*          AUTHORIZATION REQUEST.  IF THIS RESPONSE FUNCTION    *
*          FAILS TO PASS THE MESSAGE DATA TO OTM FOR PROCESSING *
*          THEN REVERSE THE MESSAGE BY STARTING EFB2 TRANSACTION.*
*                                                               *
* INPUTS:  MESSAGE LINKAGE AREA (PASS-AREA)                     *
*                                                               *
* OUTPUTS:                                                      *
*                                                               *
* ERRORS:  (RETURN CODES PASSED TO CWAC5300 IN 'MLA-RETURN-CODE' *
*           RETURN CODE       REASON                          *
*           -----------       ------------------------------  *
*                0            MESSAGE SENT SUCCESSFULLY       *
*                3            NO COMMAREA PASSED              *
*                                                               *
* REVISION HISTORY :                                            *
*****************************************************************
* 11/11/99 SMB000 RONO INITIAL PROGRAM
*****************************************************************
 EJECT
 ENVIRONMENT DIVISION.
 DATA DIVISION.
 WORKING-STORAGE SECTION.

*****************************************************************
*  SET WS-PROGRAM-ID HERE                                       *
*****************************************************************
 01  WS-WORKAREA.
     05  WS-PROGRAM-ID       PIC  X(08) VALUE 'CWUCB200'.
     05  FILLER              PIC  X(13) VALUE ' COMPILED ON:'.
     05  WS-WHEN-COMPILED    PIC  X(16).

*****************************************************************
*****   THE 01 FOLLOWING IS FOR USE IN DEBUGGING ONLY   **********
*****************************************************************
 01  TRACE-AREA.
     05  FILLER          PIC X(16)  VALUE '***TRACE AREA***'.
```

```
     05  WORK-PARA        PIC X(08).
*    05  TRACE-CYCLE      PIC  9(5)          VALUE ZERO.
*    05  TRACE-CTR        PIC  9(5)          VALUE ZERO.
*    05  WS-SUB1          PIC 9(05)          VALUE ZERO.
*    05  FILLER           PIC XX  VALUE '**'.
     05  PARA-ID          OCCURS 40   TIMES  PIC X(08).
*****************************************************************

 01  WS-PROCESS-AREA.
     COPY CWWL0080.
/
 01  SK-CICS-REFORMAT-AREA.
     COPY CWWL9090.
/
 01  STOP-PROCESSING-COMMAREA.
     COPY CWLL5800.
/
 01  MESSAGE-ROUTER-LINKAGE.
     COPY CWLL3300.
/
 01  SYSTEM-LOGGER-COMMAREA.
     COPY CWLL3100.

 01  COMMON-DEFINITIONS.
     COPY CWWL0020.

*****************************************************************
 LINKAGE SECTION.
*****************************************************************
 01  DFHCOMMAREA.
     03 MESSAGE-LINK-AREA.
     COPY CWLL5300.


*****************************************************************
 PROCEDURE DIVISION.
*****************************************************************
 A100-MAIN-ROUTINE.
     MOVE 'A100' TO WORK-PARA.

     MOVE WHEN-COMPILED TO WS-WHEN-COMPILED.
     INITIALIZE MESSAGE-ROUTER-LINKAGE.

     EXEC CICS HANDLE ABEND
               LABEL (PROCABEC-ABEND-HANDLER)
     END-EXEC.

     IF EIBCALEN > 0
        CONTINUE
     ELSE
        SET  MACK-BLANK             TO TRUE
        SET  MLA-RC-STOP-PROCESS    TO TRUE
        MOVE +1                     TO SLC-MESSAGE-LENGTH
        STRING 'B201: PROGRAM '  WS-PROGRAM-ID
               ', MESSAGE ' MLA-TRANS-ID
               ' WAS NOT SENT: NO COMMAREA WAS PASSED'
           DELIMITED BY SIZE INTO SLF-MESSAGE-VALUE
```

```
                WITH POINTER SLC-MESSAGE-LENGTH
           END-STRING
           PERFORM PROCABEC-LINK-TO-LOGGER
              THRU PROCABEC-LINK-TO-LOGGER-EXIT
           EXEC CICS RETURN END-EXEC
        END-IF.

*****************************************************************
* THIS PROGRAM ACTS AS A PASS THROUGH FOR THE EUB2 MESSAGE, IT'S *
* ONLY FUNCTION IS TO SATISFY THE INTERNAL AFC ARCHITECTURE. SO  *
* LETS MOVE THE EUB2 MESSAGE TO THE MLA OUTPUT AND PASS IT ON.   *
*****************************************************************
        SET  MLA-RC-GOOD           TO TRUE
        SET  MACK-BLANK            TO TRUE
        MOVE MLA-INPUT-TRANSACTION TO MLA-OUTPUT-TRANSACTION
        MOVE MLA-INPUT-MESSAGE-ID  TO MLA-OUTPUT-MESSAGE-ID

        COMPUTE  MLA-RETURN-TRANS-LEN =
                                     MLA-OUTPUT-DATA-LENGTH
                                   + LENGTH OF MLA-OUTPUT-HEADER.

 A100-END-PROGRAM.
*****************************************************************
*   THIS ROUTINE IS PROCESSED TO TRANSFER CONTROL TO THE OUTPUT  *
*   TRANSACTION MONITOR (OTM)                                    *
*****************************************************************
        MOVE 'A100 END' TO WORK-PARA.
            MOVE +1                       TO SLC-MESSAGE-LENGTH
            MOVE SPACES                   TO SLF-MESSAGE-VALUE
            STRING 'B210:'
                   ' READY TO START OTM '
                   ' PARA: '          WORK-PARA
                   ' MLA INPUT MSG: '   MLA-INPUT-TRANSACTION
                     DELIMITED BY SIZE
                          INTO SLF-MESSAGE-VALUE
                WITH POINTER SLC-MESSAGE-LENGTH
              PERFORM PROCABEC-LINK-TO-LOGGER
                 THRU PROCABEC-LINK-TO-LOGGER-EXIT

        SET CD-OTM      TO TRUE.
        EXEC CICS XCTL
             PROGRAM  (CD-NAMED-PGM)
             COMMAREA (MESSAGE-LINK-AREA)
             LENGTH   (LENGTH OF MESSAGE-LINK-AREA)
             RESP     (WS-RESP)
        END-EXEC.

*--- IF WE FAIL THEN REVERSE THE MESSAGE ---*
        IF WS-RESP NOT = DFHRESP (NORMAL)
           EXEC CICS START TRANSID('EFB2')
                           FROM   (MESSAGE-LINK-AREA)
                           RESP   (WS-RESP)
           END-EXEC
           IF WS-RESP NOT = DFHRESP(NORMAL)
              MOVE WS-RESP TO WS-RESP-D
              MOVE +1      TO SLC-MESSAGE-LENGTH
              STRING 'B201: '
```

```
                    'START OF THE BENEFIT REVERSAL PROG. (EFB2) '
                    'FAILED WHILE ATTEMPTING TO REVERSE A MESSAGE.'
                    ' RESP = ' WS-RESP-D
                    ' THE MESSAGE WAS NOT REVERSED!'
                 DELIMITED BY SIZE INTO SLF-MESSAGE-VALUE
                      WITH POINTER SLC-MESSAGE-LENGTH
              END-STRING
              PERFORM PROCABEC-LINK-TO-LOGGER
                 THRU PROCABEC-LINK-TO-LOGGER-EXIT
           END-IF
        END-IF
        EXEC CICS RETURN END-EXEC
        .
     A100-END-PROGRAM-EXIT.
        EXIT.


      ****************************************************************
      * COMMON ROUTINE COPYBOOKS
      ****************************************************************

-INC CWXL9920                                                          %
-INC CWXL9900                                                          %
-INC CWPL9090                                                          %

     TRACE-IT.

      *    IF  PARA-NDX > 39
      *        ADD 1           TO TRACE-CYCLE
      *        SET PARA-NDX    TO 1
      *    ELSE
      *        SET PARA-NDX UP BY 1
      *    END-IF.
      *
      *    MOVE WORK-PARA      TO PARA-ID (PARA-NDX)
           MOVE WORK-PARA      TO PARA-ID (1)
           .
     TRACE-IT-EXIT.
           EXIT.
```

□

APPENDIX G

```
*****************************************************************
 IDENTIFICATION DIVISION.
*****************************************************************
 PROGRAM-ID.    CWUBB100.
 AUTHOR.        CUBIC/CTS.
 INSTALLATION.
 DATE-WRITTEN.  JANUARY 2000.
 DATE-COMPILED.
*****************************************************************
*                                                               *
* PROGRAM NAME: CREATE INDICES FILE FOR BENEFITS TABLE          *
* PROGRAM-ID:   CWUBB100.                                       *
*                                                               *
* SYSTEM:  9121-490, MVS/ESA, CICS, COBOL II, ORACLE            *
* PROJECT: 170-2719, ELECTRONIC BENEFITS DISTRIBUTION SYSTEM    *
* DESC:    INDICES FILE USED FOR THE BENEFITS VIEW SCREEN       *
*                                                               *
* INPUTS:                                                       *
*     BENEFITS           ORACLE TABLE                           *
*                                                               *
* OUTPUTS:                                                      *
*     DEFMFB1            BENEFTIS INDICES                       *
*                                                               *
* ERRORS:                                                       *
*                                                               *
* REVISION HISTORY:                                             *
*                                                               *
*****************************************************************
* 01/25/00 SMB000 RONO INITIAL PROGRAM
*****************************************************************
/*****************************************************************
 ENVIRONMENT DIVISION.
*****************************************************************

 CONFIGURATION SECTION.

 SPECIAL-NAMES. C01 IS TOP-OF-PAGE.

 SOURCE-COMPUTER. ES-9000.

 OBJECT-COMPUTER. ES-9000.

 INPUT-OUTPUT SECTION.
 FILE-CONTROL.

     SELECT BENEFITS-INDICES-FILE        ASSIGN TO DEFMFB1
            ORGANIZATION INDEXED
            ACCESS       DYNAMIC
            RECORD KEY   MFB1-KEY
            FILE STATUS  MFB1-STATUS.

/*****************************************************************
 DATA DIVISION.
*****************************************************************

 FILE SECTION.
```

```
 FD  BENEFITS-INDICES-FILE.
 01  MFB1-RECORD.
     05  MFB1-KEY                    PIC  X(30).
     05  FILLER                      PIC  X(10).

/
 WORKING-STORAGE SECTION.
*
 01  MFB1-STATUS-FLAGS.
     05  MFB1-STATUS                 PIC  9(02).
         88  MFB1-SUCCESSFUL-IO                  VALUE 00.
         88  MFB1-DUPLICATE                      VALUE 22.
         88  MFB1-NOT-FOUND                      VALUE 23.
         88  MFB1-NULL-FILE                      VALUE 35.
         88  MFB1-END-OF-FILE                    VALUE 10.
     05  MFB1-OPEN-FLAG              PIC  9(01)  VALUE 0.
         88  MFB1-FILE-NOT-OPEN                  VALUE 0.
         88  MFB1-FILE-OPEN-I-O                  VALUE 1.
         88  MFB1-FILE-OPEN-OUTPUT               VALUE 2.
*
 01  BATCH-RETURN-CODE.
     COPY  CWWL9450.

*
 01  BATCH-RETURN-CODE.
     COPY  CWWL9450.

/
 01  MISC-FIELDS.
     05  WS-DT-TM.
       10  WS-DT-BYTE      OCCURS  7  TIMES  PIC X.
     05  WS-BIN-2N                          PIC S9(4) COMP.
     05  WS-BIN-2X   REDEFINES  WS-BIN-2N.
       10  WS-B2-HI                         PIC X.
       10  WS-B2-LO                         PIC X.
     05  WS-GMT                             PIC S9(8) COMP.
     05  WS-CCYYMMDD-HHMISS.
       10  WS-DT-CC                         PIC 99.
       10  WS-DT-YY                         PIC 99.
       10  WS-DT-MM                         PIC 99.
       10  WS-DT-DD                         PIC 99.
       10  WS-DT-HH                         PIC 99.
       10  WS-DT-MI                         PIC 99.
       10  WS-DT-SS                         PIC 99.
     05  WS-DT-TM-N   REDEFINES WS-CCYYMMDD-HHMISS  PIC 9(14).
     05  WS-SERIAL-NUM.
       10  WS-SN-HI-2                       PIC S9(4)  COMP.
       10  WS-SN-LO-4                       PIC S9(8)  COMP.

 01  ACCEPT-PARMS.
     05  AP-SERIAL-NUM.
       10  AP-SN-HI-5                       PIC 9(5).
       10  AP-SN-LO-10                      PIC 9(10).

 01  DISPLAY-VARIABLES.
     05  D-MFG-SERIAL-NUM                   PIC X(10).
     05  D-EMP-NAME    PIC X(10).
```

```
           05  D-SALARY       PIC Z(4)9.99.
           05  D-COMMISSION   PIC Z(4)9.99.
           05  D-INITIAL-VAL-AMT                PIC Z(2)9.99.
           05  D-SQLCODE                        PIC -9(5).

           EXEC SQL BEGIN DECLARE SECTION END-EXEC.
      *
      *  BENEFITS COLUMN DEFINITIONS
      *
-INC CWELB100                                                          %
           EXEC SQL END DECLARE SECTION END-EXEC.

           EXEC SQL INCLUDE SQLCA END-EXEC

           EXEC SQL    DECLARE ALL_BENEFITS  CURSOR FOR
               SELECT
-INC CWELB800                                                          %
               FROM MCHECK.BENEFITS
           END-EXEC
      /
       PROCEDURE DIVISION.

       A000-CONTROL.
           DISPLAY 'A000:       '
           EXEC SQL WHENEVER SQLERROR
               DO PERFORM Z300-SQL-ERROR END-EXEC
           DISPLAY '            STEP 1'

           EXEC SQL OPEN ALL_BENEFITS  END-EXEC
           MOVE     SQLCODE                TO  D-SQLCODE
           DISPLAY 'OPEN SQLCODE: '  D-SQLCODE

           PERFORM  B100-INITIALIZE THRU
                    B100-INITIALIZE-EXIT

           PERFORM  C100-CREATE-INDICES THRU
                    C100-CREATE-INDICES-EXIT
               UNTIL  SQLCODE  =  +100
                  OR  SQLCODE  =  +1403

           PERFORM  Z100-SIGN-OFF THRU
                    Z100-SIGN-OFF-EXIT
           .
       A000-CONTROL-EXIT.
           EXIT.
           GOBACK.

       B100-INITIALIZE.
           DISPLAY 'B100:          '
           .
       B100-INITIALIZE-EXIT.
           EXIT.

       C100-CREATE-INDICES.
           DISPLAY 'C100:          '

      *  FETCH BENEFITS ROW
```

```
      *  WRITE INDEX FILE

           EXEC SQL FETCH   ALL_BENEFITS
              INTO
-INC CWELB805                                                          %
           END-EXEC

           DISPLAY  BEN-SERIAL-NUM
              '-*-' BEN-CUSTOMER-ID
              '-*-' BEN-BENEFIT-TYPE
              '-*-' BEN-EFFECTIVE-DATE
           .
       C100-CREATE-INDICES-EXIT.
           EXIT.

      /*****************************************************************
      ***  THIS OPENS THE INDECES FILE
      *****************************************************************

       V015-OPEN-MFB1.
      *****************************************************************

           DISPLAY 'V015:'.

           OPEN    I-O            BENEFITS-INDICES-FILE
           EVALUATE TRUE
             WHEN MFB1-SUCCESSFUL-IO
               SET  MFB1-FILE-OPEN-I-O     TO  TRUE
             WHEN MFB1-NULL-FILE
               OPEN  OUTPUT        BENEFITS-INDICES-FILE
               EVALUATE  TRUE
                 WHEN MFB1-SUCCESSFUL-IO
                   SET  MFB1-FILE-OPEN-OUTPUT  TO  TRUE
                 WHEN OTHER
                   DISPLAY 'ERROR OPENING **OUTPUT** BENEFITS INDICES'
                           ' BEN IND FILE - '
                         'STATUS: ' MFB1-STATUS
                   SET  BRC-OPEN-ERROR TO TRUE
               END-EVALUATE
             WHEN OTHER
                 DISPLAY 'ERROR OPENING ** I-O  ** BEN INDICES '
                         'FILE - '
                       'STATUS: ' MFB1-STATUS
               SET  BRC-OPEN-ERROR TO TRUE
           END-EVALUATE
           .
       V015-OPEN-MFB1-EXIT.
           EXIT.

      *****************************************************************
      ***  CREATES DUMMY RECORD FOR BENEFITS INDECES FILE
      *****************************************************************

       V018-CREATE-DUMMY-MFB1.
      *****************************************************************

           DISPLAY 'V018:'.
```

```
     OPEN    OUTPUT               BENEFITS-INDICES-FILE
     EVALUATE TRUE
       WHEN MFB1-SUCCESSFUL-IO
         CONTINUE
       WHEN OTHER
         DISPLAY 'DUMMY RECORD PROCESSING.  '
         DISPLAY 'ERROR OPENING BENEFITS INDICES  FILE - '
                 'STATUS: ' MFB1-STATUS
         SET  BRC-OPEN-ERROR TO TRUE
         GO TO V018-CREATE-DUMMY-MFB1-EXIT
     END-EVALUATE

     MOVE  LOW-VALUES             TO  MFB1-RECORD
                                      WS-MFB1-KEY
     MOVE  'DUMMY'                TO  WS-MFB1-SET-ID
     MOVE  WS-MFB1-KEY            TO  MFB1-KEY
     WRITE   MFB1-RECORD
     EVALUATE TRUE
       WHEN MFB1-SUCCESSFUL-IO
         CONTINUE
       WHEN OTHER
         DISPLAY 'DUMMY RECORD PROCESSING.  '
         DISPLAY 'V018: ERROR WRITING BENEFITS INDICES  FILE - '
                 'STATUS: ' MFB1-STATUS
                 ' BEN INDICES:  KEY: '
                    MFB1-KEY
         SET  BRC-OPEN-ERROR TO TRUE
         GO TO V018-CREATE-DUMMY-MFB1-EXIT
     END-EVALUATE
     CLOSE                        BENEFITS-INDICES-FILE
     .
 V018-CREATE-DUMMY-MFB1-EXIT.
     EXIT.
/*****************************************************************
***  THIS READS BENEFITS INDICES  FILE TO DETERMINE UPDATE/WRITE
*****************************************************************

 V030-UPDATE-BEN-INDICES.
*****************************************************************

     EVALUATE TRUE
         WHEN MFB1-FILE-OPEN-I-O
             READ  BENEFITS-INDICES-FILE
             EVALUATE TRUE
              WHEN  MFB1-SUCCESSFUL-IO
               PERFORM  V033-REWRITE-BEN-INDICES THRU
                        V033-REWRITE-BEN-INDICES-EXIT
              WHEN  MFB1-NOT-FOUND
               PERFORM  V035-WRITE-BEN-INDICES THRU
                        V035-WRITE-BEN-INDICES-EXIT
              WHEN  OTHER
               DISPLAY 'ERROR READING BENEFITS INDICES  FILE - '
                       'STATUS: ' MFB1-STATUS
                       ' 1-30: '   BEN-INDICES   (1 : 20)
               SET  BRC-IO-ERROR TO TRUE
```

```
            END-EVALUATE
        WHEN MFB1-FILE-OPEN-OUTPUT
            PERFORM  V035-WRITE-BEN-INDICES THRU
                     V035-WRITE-BEN-INDICES-EXIT
        WHEN OTHER
            DISPLAY 'ERROR READING FARE BEN INDICES  FILE - '
                    'STATUS: ' MFB1-STATUS
                    ' 1-20: '   BEN-INDICES (1 : 20)
            SET  BRC-IO-ERROR TO TRUE
    END-EVALUATE
    .
 V030-UPDATE-BEN-INDICES-EXIT.
    EXIT.
/*****************************************************************
***  REWRITES BENEFITS INDICES RECORD
*****************************************************************

 V033-REWRITE-BEN-INDICES.
*****************************************************************

    REWRITE  MFB1-RECORD      FROM  BEN-INDICES
    EVALUATE TRUE
        WHEN MFB1-SUCCESSFUL-IO
            ADD  1            TO WS-MFB1-WRITE
        WHEN OTHER
            DISPLAY 'ERROR RE-WRITING BENEFITS INDICES  FILE - '
                    'STATUS: '  MFB1-STATUS
                    ' BEN IND TABLE:  KEY: '
                   MFB1-KEY
            SET  BRC-IO-ERROR TO TRUE
    END-EVALUATE
    .
 V033-REWRITE-BEN-INDICES-EXIT.
    EXIT.
/*****************************************************************
***  WRITES NEW BENEFITS INDICES RECORD
*****************************************************************
 V035-WRITE-BEN-INDICES.
*****************************************************************

    WRITE  MFB1-RECORD      FROM  BEN-INDICES
    EVALUATE TRUE
        WHEN MFB1-SUCCESSFUL-IO
            ADD  1            TO WS-MFB1-WRITE
        WHEN OTHER
            DISPLAY 'ERROR  WRITING BENEFITS INDICES FILE - '
                    'STATUS: ' MFB1-STATUS
                    ' BEN INDICES:  KEY: '
                   MFB1-KEY
            SET  BRC-IO-ERROR TO TRUE
    END-EVALUATE
    .
 V035-WRITE-BEN-INDICES-EXIT.
    EXIT.
/*****************************************************************
***  THIS CLOSES THE BENEFITS INDICES FILE
*****************************************************************
```

```
 V070-CLOSE-MFB1.
*****************************************************************

     DISPLAY 'V070:'.

     CLOSE                          BENEFITS-INDICES-FILE
     EVALUATE TRUE
         WHEN MFB1-SUCCESSFUL-IO
             CONTINUE
         WHEN OTHER
             DISPLAY 'ERROR CLOSING BENEFITS INDICES FILE - '
                     'STATUS: ' MFB1-STATUS
             SET  BRC-CLOSE-ERROR TO TRUE
     END-EVALUATE.

 V070-CLOSE-MFB1-EXIT.
     EXIT.
/
 Z100-SIGN-OFF.
     EXEC SQL CLOSE ALL_BENEFITS   END-EXEC
     DISPLAY ' '
     DISPLAY 'HAVE A GOOD DAY!'
     DISPLAY ' '
     EXEC SQL COMMIT WORK RELEASE END-EXEC
     GOBACK
     .
 Z100-SIGN-OFF-EXIT.
     EXIT.

 Z300-SQL-ERROR.
     MOVE     SQLCODE                 TO  D-SQLCODE
     EXEC SQL WHENEVER SQLERROR CONTINUE END-EXEC
     DISPLAY ' '
     DISPLAY 'ORACLE ERROR DETECTED: '  D-SQLCODE
     DISPLAY ' '
     DISPLAY SQLERRMC
     EXEC SQL ROLLBACK WORK RELEASE END-EXEC
     GOBACK
     .
 Z300-SQL-ERROR-EXIT.
     EXIT.
```

APPENDIX H

```
*****************************************************************00000100
 IDENTIFICATION DIVISION.                                        00000200
*****************************************************************00000300
 PROGRAM-ID.    CWOCB100.                                        00000400
 AUTHOR.        CUBIC/CTS.                                       00000500
 INSTALLATION.                                                   00000600
 DATE-WRITTEN.  JANUARY 2000.                                    00000700
 DATE-COMPILED.                                                  00000800
*****************************************************************00000900
*                                                               *00001000
* PROGRAM NAME: BENEFITS VIEW/MAINTENANCE                       *00001100
* PROGRAM ID:   CWOCB100                                        *00001200
*                                                               *00001300
*                                                               *00001400
* SYSTEM:  9121-490, MVS/XA,  CICS, COBOL II, ORACLE            *00001500
* PROJECT: 170-2719, ELECTRONIC BENEFITS DISTRIBUTION SYSTEM    *00001600
*                                                               *00001700
*                                                               *00001800
* DESC:     THIS PROGRAM ALLOWS THE USER TO VIEW                *00001900
*           BENEFITS INFORMATION.  BENEFITS CAN BE VIEWED BY    *00002000
*           MANUFACTURER SERIAL NUMBER OR CUSTOMER ID.          *00002100
*                                                               *00002300
*                                                               *00002400
* INPUTS:   - STANDARD COMMAREA                                 *00002500
*           - USER COMMAREA ATTACHED TO STANDARD COMMAREA       *00002600
*             (CWLLB100  COPYBOOK)                              *00002700
*           - BENEFITS TABLE (ORACLE)                           *00002800
*                                                               *00003100
* OUTPUTS:  - NONE                                              *00003200
*                                                               *00003300
* ERRORS:   - SEE WS-ERROR-MESSAGES FOR A LIST OF ERRORS PUT OUT *00003400
*             BY THIS PROGRAM                                   *00003500
*                                                               *00003600
*****************************************************************00003700
* REVISION HISTORY:                                             *00003800
*****************************************************************00003700
* MM/DD/YY XX9999 XXXX DESCRIPTION                              *00004000
*                                                               *00003900
*                                                               *00003900
*                                                               *00003900
*****************************************************************00005400
/                                                                00005500
 DATA DIVISION.                                                  00005600
                                                                 00005700
 WORKING-STORAGE SECTION.                                        00005800
                                                                 00005900
 01  WS-ID-AREAS.                                                00006000
     05  FILLER                      PIC  X(16)  VALUE           00006100
         'START OF WS FOR '.                                     00006200
     05  WS-PROGRAM-ID               PIC  X(08)  VALUE 'CWOCB100'.00006300
     05  FILLER                      PIC  X(12)  VALUE           00006400
         ' COMPILE ON '.                                         00006500
     05  WS-WHEN-COMPILED            PIC  X(16)  VALUE SPACES.   00006600
     05  FILLER                      PIC  X(04)  VALUE ' ==>'.   00006700
     05  WS-BMS-MAP-SET              PIC  X(08)  VALUE 'CWOB100 '.00006800
     05  WS-BMS-MAP                  PIC  X(08)  VALUE 'CWOB100 '.00006900
     05  WS-NEXT-PGM-DOWN            PIC  X(08)  VALUE 'CWOCB100'.00007000
```

```
                                                                        00007100
 01  WS-TRACE-AREA.                                                    00007200
     05  FILLER                          PIC  X(16)  VALUE             00007300
         '***TRACE AREA***'.                                            00007400
     05  WORK-PARA                       PIC  X(08).                   00007500
     05  TRACE-CYCLE                     PIC  9(05)  VALUE ZEROES.     00007600
     05  FILLER                          PIC  X(02)  VALUE '**'.       00007700
     05  PARA-ID                         PIC  X(08).                   00007800
/                                                                       00008100
 01  WS-PROCESS-AREA.                                                  00008200
     COPY CWWL0080.                                                     00008300
/                                                                       00008100
 01  SKLEAST-AREA.                                                     00008200
     COPY CWWL5200.                                                     00008300
                                                                        00008400
 01  SK-MAX-QUEUE                    PIC  9(02)  VALUE 1.              00008500
/                                                                       00008600
 01  RF-REFERENCE-RCD.                                                 00008700
     COPY CWVL0300.                                                     00008800
/                                                                       00008900
 01  RF-TABLE-ID-VALUES.                                               00009000
     COPY CWWL0388.                                                     00009100
/                                                                       00009500
 01  WS-WORK-AREAS.                                                    00009600
     05  WS-DISP-NUM-5               PIC  Z(4)9.                       00013400
     05  WS-DISP-NUM-10              PIC  Z(9)9.                       00013400
     05  WS-DISP-DEC-FMT-5           PIC  Z(02)9.99.                   00013400
     05  WS-DISP-ZZ9                 PIC ZZ9.                          00013400
     05  WS-BIN-3X.                                                    00010400
         10  WS-BIN-3X-HI            PIC  X(01)  VALUE LOW-VALUES.     00010500
         10  WS-BIN-3X-LO            PIC  X(03).                       00010600
     05  WS-BIN-3C                                                     00010700
         REDEFINES WS-BIN-3X          PIC S9(08)  COMP.                 00010800

     05  WS-BIN-2X.                                                    00010900
         10  WS-BIN-2X-HI            PIC  X(01)  VALUE LOW-VALUES.     00011000
         10  WS-BIN-2X-LO            PIC  X(01).                       00011100
     05  WS-BIN-2C                                                     00011200
         REDEFINES WS-BIN-2X          PIC S9(04)  COMP.                 00011300
                                                                        00013700
     05  WS-USERID                   PIC X(08) VALUE SPACES.           00013700
                                                                        00013700
 01  WS-PARTICULAR-AREAS.                                              00013800
     05  WS-GET-ROW-FLAG             PIC  9.                           00014200
         88  WS-GET-NEXT-ROW                     VALUE 1.              00014400
         88  WS-GET-PREVIOUS-ROW                 VALUE 3.              00014500
         88  WS-GET-CURRENT-ROW                  VALUE 5.              00014600


                                                                        00013700
     05  WS-SAVE-MFB1-KEY            PIC  X(68).                       00014200
                                                                        00013700
     05  WS-SEARCH-FLAG              PIC  9      VALUE 0.              00014200
         88  WS-GOOD-SEARCH                      VALUE 0.
         88  WS-END-SEARCH                       VALUE 1.
                                                                        00013700
     05  WS-FUNCTION-ID              PIC  X(8).                        00014200
         88  WS-STARTBR          VALUE 'STARTBR '.
```

```
           88  WS-READNEXT          VALUE 'READNEXT'.
           88  WS-READPREV          VALUE 'READPREV'.


       05  WS-DESC                      PIC  X(15).                  00017800

       05  WS-FORMAT-DATE.                                           00019100
           10  WS-FORMAT-DATE-YYYY      PIC  9(04).                  00019200
           10  FILLER                   PIC  X(01)  VALUE '/'.       00019300
           10  WS-FORMAT-DATE-MM        PIC  9(02).                  00019400
           10  FILLER                   PIC  X(01)  VALUE '/'.       00019500
           10  WS-FORMAT-DATE-DD        PIC  9(02).                  00019600
       05  WS-FORMAT-TIME.                                           00019700
           10  WS-FORMAT-TIME-HH        PIC  9(02).                  00019800
           10  FILLER                   PIC  X(01)  VALUE ':'.       00019900
           10  WS-FORMAT-TIME-MM        PIC  9(02).                  00020000
           10  FILLER                   PIC  X(01)  VALUE ':'.       00020100
           10  WS-FORMAT-TIME-SS        PIC  9(02).                  00020200

/                                                                    00021000
 01  WS-PFKEY-MESSAGES.                                             00021100
       05  WS-PFKEY-MSG1.                                           00021200
           10  FILLER                   PIC  X(40)  VALUE          00021300
               'PF4=UPDATE 9=CARDBACK 10=CARDFWD      '.            00021400
           10  FILLER                   PIC  X(20)  VALUE          00021500
               '                    '.                             00021600
       05  WS-PFKEY-MSG2.                                           00021700
           10  FILLER                   PIC  X(33)  VALUE          00021800
               'PF2=SET HOLD 6=BROWSE             '.                00021900
           10  FILLER                   PIC  X(20)  VALUE          00022000
               '                    '.                             00021600
       05  WS-PFKEY-MSG3.                                           00021700
           10  FILLER                   PIC  X(33)  VALUE          00021800
               'PF5=CONFIRM 6=BROWSE              '.                00021900
           10  FILLER                   PIC  X(20)  VALUE          00022000
               '                    '.                             00022100
                                                                    00022200
/                                                                   00022400
*01  BIT-MASK-VALUES.                                               00022500
       COPY CWWL9410.                                               00022600
/                                                                   00022700
 01  WS-ERROR-MESSAGES.                                             00022800
       05  WS-B101-NUM                  PIC  X(04)  VALUE 'B101'.   00022900
       05  WS-B101-MSG                  PIC  X(63)  VALUE          00023000
           'ENTER CARD SER#                                      '. 00023100
                                                                    00022200
       05  WS-B102-NUM                  PIC  X(04)  VALUE 'B102'.   00022900
       05  WS-B102-MSG                  PIC  X(63)  VALUE          00023000
           'NOW YOU CAN TOGGLE THE HOLD ON THIS ROW              '. 00023100
                                                                    00022200
       05  WS-B103-NUM                  PIC  X(04)  VALUE 'B103'.   00022900
       05  WS-B103-MSG                  PIC  X(63)  VALUE          00023000
           'YOU ARE NOW IN THE BROWSE MODE                       '. 00023100
                                                                    00022200
       05  WS-B105-NUM                  PIC  X(04)  VALUE 'B105'.   00022900
       05  WS-B105-MSG                  PIC  X(63)  VALUE          00023000
           'PRESS PF5 TO COMMIT CHANGE                           '. 00023100
```

```
                                                                00022200
05  WS-B115-NUM                 PIC  X(04)  VALUE 'B115'.     00022900
05  WS-B115-MSG.                                               00023000
  10  WS-B115-SERIAL-NUM         PIC  X(15).                   00023000
  10  FILLER                     PIC  X      VALUE '*'.        00023000
  10  WS-B115-CUSTOMER-ID        PIC  X(14).                   00023000
  10  FILLER                     PIC  X      VALUE '*'.        00023000
  10  WS-B115-BENEFIT-TYPE       PIC  X(5).                    00023000
  10  FILLER                     PIC  X      VALUE '*'.        00023000
  10  WS-B115-EFFECTIVE-DATE     PIC  X(14).                   00023000
  10  FILLER                     PIC  X      VALUE '*'.        00023000
                                                                00022200
05  WS-B121-NUM                 PIC  X(04)  VALUE 'B121'.     00023500
05  WS-B121-MSG                 PIC  X(63)  VALUE             00023600
    'ENTRY MUST BE NUMERIC:                                  '. 00023700
                                                                00022200
05  WS-B180-NUM                 PIC  X(04)  VALUE 'B180'.     00022900
05  WS-B180-MSG.                                               00023000
  10  FILLER                     PIC  X(50)  VALUE             00023000
    '12345 ROW - END OF LIST                                 '. 00023100
  10  FILLER                     PIC  X(13)  VALUE             00023000
    '             '.                                          00023100
                                                                00022200
05  WS-B181-NUM                 PIC  X(04)  VALUE 'B181'.     00022900
05  WS-B181-MSG.                                               00023000
  10  FILLER                     PIC  X(50)  VALUE             00023000
    'SQL CODE: XXXXX PARA: 12345678                          '. 00023100
  10  FILLER                     PIC  X(13)  VALUE             00023000
    '   NEW PROG  '.                                          00023100
                                                                00022200
05  WS-B183-NUM                 PIC  X(04)  VALUE 'B183'.     00025600
05  WS-B183-MSG.                                               00025700
    10  FILLER                  PIC  X(36)  VALUE             00025800
        'UNSUCCESSFUL BEFORE IMAGE WRITE TO  '.               00025900
    10  FILLER                  PIC  X(27)  VALUE             00025800
        'SYSTLOG                    '.                        00025900
                                                                00022200
05  WS-B184-NUM                 PIC  X(04)  VALUE 'B184'.     00025600
05  WS-B184-MSG.                                               00025700
    10  FILLER                  PIC  X(36)  VALUE             00025800
        'UNSUCCESSFUL AFTER  IMAGE WRITE TO  '.               00025900
    10  FILLER                  PIC  X(27)  VALUE             00025800
        'SYSTLOG                    '.                        00025900

05  WS-B185-NUM                 PIC  X(04)  VALUE 'B185'.     00025600
05  WS-B185-MSG.                                               00025700
  10  FILLER                  PIC  X(36)  VALUE                00025800
    'ERROR DATE/TIME CONVERT. PLEASE EXIT'.
  10  FILLER                  PIC  X(27)  VALUE                00025800
    '                           '.                            00025900

05  WS-B186-NUM                 PIC  X(04)  VALUE 'B186'.     00025600
05  WS-B186-MSG.                                               00025700
  10  FILLER                  PIC  X(36)  VALUE                00025800
    'DATE PGM LINK ERROR, RESP=   .      '.
  10  FILLER                  PIC  X(27)  VALUE                00025800
      ' PLEASE EXIT.              '.                          00025900
```

```
     05  WS-B187-NUM                  PIC  X(04)  VALUE 'B187'.    00025600
     05  WS-B187-MSG.                                               00025700
       10  FILLER                   PIC  X(36)  VALUE             00025800
         'INVALID MANUAL HOLD CODE:    CODE NO'.
       10  FILLER                   PIC  X(27)  VALUE             00025800
           'T CHANGED!                   '.                        00025900

     05  WS-B188-NUM                  PIC  X(04)  VALUE 'B187'.    00025600
     05  WS-B188-MSG.                                               00025700
       10  FILLER                   PIC  X(36)  VALUE             00025800
         'UNABLE TO TOGGLE VENDOR HOLD CODE   '.
       10  FILLER                   PIC  X(27)  VALUE             00025800
           '                             '.                        00025900
                                                                   00022200
     05  WS-B189-NUM                  PIC  X(04)  VALUE 'B189'.    00022900
     05  WS-B189-MSG.                                               00023000
       10  FILLER                     PIC  X(50)  VALUE           00023000
         'VSAM ERROR RESP: XXXXX PARA: 12345678             '.     00023100
       10  FILLER                     PIC  X(13)  VALUE           00023000
         '             '.                                          00023100
                                                                   00022200
     05  WS-B190-NUM                  PIC  X(04)  VALUE 'B190'.    00022900
     05  WS-B190-MSG.                                               00023000
       10  FILLER                     PIC  X(50)  VALUE           00023000
         'INVALID SEARCH FLAG: X                              '.   00023100
       10  FILLER                     PIC  X(13)  VALUE           00023000
         '             '.                                          00023100
                                                                   00022200
     05  WS-B191-NUM                  PIC  X(04)  VALUE 'B191'.    00022900
     05  WS-B191-MSG.                                               00023000
       10  FILLER                     PIC  X(50)  VALUE           00023000
         'INVALID SEARCH FLAG: X                              '.   00023100
       10  FILLER                     PIC  X(13)  VALUE           00023000
         '             '.                                          00023100
                                                                   00022200

*** RESP CODE                        ***

     05  WS-RESP-9-DISPLAY            PIC  9(02).
/                                                                  00026400
 01  COMMON-DEFINITIONS.                                          00026500
     COPY CWWL0020.                                                00026600
/                                                                  00026700
 01  SCREEN-QUEUE-AREA.                                           00026800
     COPY CWQL5500.                                                00026900
/                                                                  00027000
 01  WS-GENERAL-DATA.                                             00027100
     05  WS-FCI                       PIC  X(01).                 00027200
         88  WS-FCI-TERM                          VALUE X'01'.    00027300
     05  WS-MAP-LEN                   PIC S9(04)  COMP            00027400
                                                 VALUE ZEROES.    00027500
     05  WS-MAP-PTR                   USAGE IS POINTER.           00027600
     05  WS-RECORD-LENGTH             PIC S9(04)  COMP.           00027700
     05  WS-ITEM                      PIC S9(04)  COMP            00027800
                                                 VALUE +1.        00027900
     05    I                           PIC S9(04)  COMP.           00027800
```

```
      /                                                                 00028200
       01  NUMERIC-CHECK-VARIABLES.                                     00028300
           COPY CWWL9020.                                               00028400
      /                                                                 00028500
       01  UNSTRING-VARIABLES.                                          00028600
           COPY CWWL9030.                                               00028700
      /                                                                 00028800
       01  STANDARD-COPIED-MESSAGES.                                    00028900
           COPY CWWL0030.                                               00029000
      /                                                                 00029100
       01  SK-CICS-REFORMAT-AREA.                                       00029200
           COPY CWWL9090.                                               00029300
      /                                                                 00029400
       01  MESSAGE-ROUTER-LINKAGE.                                      00029500
           COPY CWLL3300.                                               00029600
      /                                                                 00029700
       01  SYSTEM-LOGGER-COMMAREA.                                      00029800
           COPY CWLL3100.                                               00029900
      /                                                                 00030000
       01  DATE-TIME-CONVERSION.                                        00030100
           COPY CWLL3000.                                               00030200
      /                                                                 00030300
           COPY DFHBMSCA.                                               00030400
                                                                        00030500
           COPY DFHAID.                                                 00030600
      /                                                                 00030700
      *
       01  ORACLE-SQL-CODES.                                            00013400
-INC CWELB000                                                           %
      *
      *    SQL HOST VARIABLES                                           00030800
      /                                                                 00030900
           EXEC SQL                                                     00031000
               BEGIN DECLARE SECTION                                    00031100
           END-EXEC                                                     00031200
      *                                                                 00031000
-INC CWELB100                                                           %
      *                                                                 00031000
-INC CWELB101                                                           %
      *
       01  BENEFITS-DEFINITION-ROW.                                     00013400
         05  BND-BENEFIT-DESC                PIC   X(60) VARYING.       00013500
      *
       01  WS-SQL-HOST-VARIABLES.                                       00013400
         05  WS-PROCESS-DATE                 PIC   X(7).                00013500
         05  WS-PROCESS-DATE-X14             PIC   X(14).               00013500
         05  WS-PD-X14  REDEFINES WS-PROCESS-DATE-X14.                  00013500
           10  WS-PD-DATE.                                             00013500
             15  WS-PD-CC                      PIC  S99.               00013500
             15  WS-PD-YY                      PIC  S99.               00013500
             15  WS-PD-MM                      PIC  S99.               00013500
             15  WS-PD-DD                      PIC  S99.               00013500
           10  WS-PD-TIME.                                             00013500
             15  WS-PD-HH                      PIC  S99.               00013500
             15  WS-PD-MI                      PIC  S99.               00013500
             15  WS-PD-SS                      PIC  S99.               00013500
         05  WS-USER-ID                      PIC   X(8).                00014000
```

```
                                                                        00013300
     EXEC SQL VAR
       WS-PROCESS-DATE         IS  DATE
     END-EXEC
                                                                        00013300
     EXEC SQL
       END DECLARE SECTION
     END-EXEC

     EXEC SQL
       INCLUDE SQLCACOB
     END-EXEC

*    EXEC SQL
*      INCLUDE SQLCA
*    END-EXEC

     EXEC SQL
       INCLUDE ORACA
     END-EXEC

 01  WS-MISC-FIELDS.
   05  WS-SQLCODE-DISP                      PIC -(4)9.
   05  WS-SQLCODE-C                         PIC X(9).
   05  WS-SQLCODE REDEFINES WS-SQLCODE-C    PIC 9(9).
   05  HV-SERIAL-NUM                        PIC X(15).
     88  HV-FIRST-SER-NUM         VALUE '000000000000001'.
   05  HV-MFG-SERIAL-NUM                    PIC X(11).
     88  HV-FIRST-MFG-SER-NUM     VALUE '00000000001'.
   05  HV-CUSTOMER-ID                       PIC X(14).
     88  HV-FIRST-CUST-ID         VALUE 'AAAAAAAAAAAAAA'.

 01  WS-MFB1-RECORD.
     COPY CWVLB100.

 01  WS-QUEUE-DATA.
     05  WS-SCREEN-LOGIC-DATA.
         10  WS-ORDER-FLAG           PIC  9(01).
             88  WS-SERIAL-ORDER                 VALUE 1.
             88  WS-NAME-ORDER                   VALUE 2.
             88  WS-MFG-ORDER                    VALUE 3.
                                                                        00050500
/                                                                       00052500
 LINKAGE SECTION.                                                       00052600
                                                                        00052700
 01  DFHCOMMAREA.                                                       00052800
     COPY CWLL0000.                                                     00052900
/                                                                       00053000
     COPY CWLLB100.                                                     00053100
/                                                                       00053200
                                                                        00053200
*** SCREEN CWOB100 COPYBOOK.                                           00053200
                                                                        00053200
 COPY CWOB100.                                                          00053300
/                                                                       00053400
/                                                                       00053600
 01  GETMAIN-AREA                    PIC  X(8000).                     00053700
```

```
                                                                        00053800
       01  BIG-AREA                          PIC  X(4000).             00053900
      /                                                                 00054000
       PROCEDURE DIVISION.                                              00054100
       A000-CONTROL-PROCESS.                                            00054200
                                                                        00054300
           MOVE WHEN-COMPILED       TO WS-WHEN-COMPILED.                00054900
                                                                        00055000
           EXEC CICS ASSIGN                                             00055100
                     FCI (WS-FCI)                                       00055200
           END-EXEC.                                                    00055300
                                                                        00055400
           IF  NOT WS-FCI-TERM                                          00055500
               GO TO SKABEXIT-NO-TERM                                   00055600
           END-IF.                                                      00055700
                                                                        00055800
           IF  EIBCALEN = ZEROES                                        00055900
               GO TO SKABEXIT-NO-COMMAREA                               00056000
           END-IF                                                       00056100
                                                                        00056200
                                                                        00031500
           SET  ADDRESS OF CWOB100O     TO    STC-MAP-PTR               00031600
           SET  WS-MAP-PTR              TO    STC-MAP-PTR               00031700
           SET  SQA-EXTRA-QUE   TO TRUE                                 00056500
                                                                        00058500
           MOVE SPACES                  TO MSG1O                        00059600
                                           MSG2O                        00059600
                                           MSG3O                        00059700
                                                                        00056700
           EVALUATE TRUE                                                00056800
               WHEN STC-NEXT-SCREEN                                     00056900
                    PERFORM B000-INITIAL-SEND                           00057000
                       THRU B000-INITIAL-SEND-EXIT                      00057100
                                                                        00057700
               WHEN STC-GOING-LATERAL                                   00057800
                    PERFORM F000-LATERAL-PROCESS                        00057900
                       THRU F000-LATERAL-PROCESS-EXIT                   00058000
                                                                        00058100
               WHEN OTHER                                               00058200
                    GO TO SKABEXIT-LEVEL2                               00058300
           END-EVALUATE                                                 00058400
                                                                        00060300
SMB000     EXEC  SQL                                                    00013400
              COMMIT  WORK  RELEASE                                     00013400
           END-EXEC                                                     00013400
                                                                        00060300
           MOVE WS-PROGRAM-ID       TO PROGNAMO                         00059800
           MOVE MSGNUMO             TO STC-HELP-MSG                     00059900
           MOVE WS-BMS-MAP-SET      TO STC-CURR-MAPSET                  00060000
           MOVE WS-BMS-MAP          TO STC-CURR-MAP                     00060100
           MOVE LENGTH OF CWOB100O  TO STC-MAP-LEN                      00060200
                                                                        00060600
           EXEC CICS XCTL                                               00060700
             PROGRAM   (CD-NAMED-PGM)                                   00060800
             COMMAREA  (DFHCOMMAREA)                                    00060900
             LENGTH    (EIBCALEN)                                       00061000
             RESP      (WS-RESP)                                        00061100
```

```
     END-EXEC                                                          00061200
                                                                       00061300
     IF  WS-RESP NOT = DFHRESP (NORMAL)                                00061400
         GO TO SKABEXIT-LEVEL2                                         00061500
     END-IF                                                            00061600
                                                                       00061700
     EXEC CICS RETURN                                                  00061800
     END-EXEC.                                                         00061900
/                                                                      00062000
 B000-INITIAL-SEND.                                                    00062100
*****************************************************************    00062200
*  THIS PARAGRAPH WILL PERFORM ANY INITIALIZATION NEEDED WHEN   *      00062300
*  THIS PROGRAM IS INVOKED FOR THE FIRST TIME.                  *      00062400
*****************************************************************    00062500
                                                                       00062600
     MOVE 'B000' TO WORK-PARA.                                         00062700
                                                                       00062900
     COMPUTE STC-MAX-QUEUE = STC-MAX-QUEUE + SK-MAX-QUEUE              00063000
                                                                       00062900
                                                                       00062900
     EXEC SQL SELECT
           USER INTO :WS-USER-ID
         FROM  DUAL
     END-EXEC
                                                                       00062900
     MOVE  WS-USER-ID            TO  STU-USER-ID
                                                                       00063100
     MOVE WS-B101-NUM            TO MSGNUMO                            00063200
                                    STC-HELP-MSG                       00063300
     MOVE WS-B101-MSG            TO MSG1O                              00063400
                                                                       00065100
     SET STU-INITIAL-SCREEN  TO TRUE                                   00065100
                                                                       00065100
     MOVE  -1                    TO  CARD-SER1L                        00065200
                                                                       00065100
     SET  CD-SCR-ROUTER-EXIT                                           00066400
          STC-EDIT-INITIAL                                             00066600
          STC-GOING-LATERAL                                            00066600
          STC-FCN-NA         TO  TRUE
     .
 B000-INITIAL-SEND-EXIT.                                               00066900
     EXIT.                                                             00067000
/                                                                      00067100
 F000-LATERAL-PROCESS.                                                 00078600
*****************************************************************    00078700
*  THIS PARAGRAPH IS PERFORMED TO PROCESS USER INPUT.           *      00078800
*****************************************************************    00078900
                                                                       00079000
     MOVE 'F000' TO WORK-PARA                                          00079100
                                                                       00079300
     MOVE ZEROES                          TO MSGNUMO                   00079400
                                             STC-HELP-MSG              00063300
     MOVE SPACES                          TO MSG1O                     00079500
                                                                       00080000
     SET  WS-EDIT-OK                      TO TRUE                      00079600

     EVALUATE  TRUE                                                    00079100
```

```
WHEN  STU-INITIAL-SCREEN                                          00079100
  SET  STU-SEARCH-SER-NUM TO  TRUE
  SET  STU-PREV-KEY-ENTER  TO  TRUE                               00079100
  MOVE  WS-PFKEY-MSG1                TO  PFKEY-MSGO               00079100
  PERFORM  M201-EDIT-SER-NUM THRU                                 00079100
           M201-EDIT-SER-NUM-EXIT                                 00079100
  IF  WS-EDIT-OK
    MOVE  HV-SERIAL-NUM              TO  BEN-SERIAL-NUM
    MOVE  '00000000000001'           TO  BEN-CUSTOMER-ID
    PERFORM  F100-DISP-NEXT-ROW THRU
             F100-DISP-NEXT-ROW-EXIT
    IF  WS-EDIT-OK                                                00079100
      SET  STU-BROWSE       TO  TRUE                            00079100
    END-IF
  END-IF

WHEN  STU-BROWSE                                                  00079100

  MOVE  STU-BEN-ROW                TO  BENEFITS-ROW

  MOVE  WS-PFKEY-MSG1                TO  PFKEY-MSGO               00079100
  PERFORM  M200-EDIT-SEARCH-KEY THRU                              00079100
           M200-EDIT-SEARCH-KEY-EXIT                              00079100
  IF  WS-EDIT-ERROR
    GO TO  F000-LATERAL-PRE-EXIT
  END-IF

  IF    BEN-SERIAL-NUM       =  HV-SERIAL-NUM                     00079100
   AND  BEN-MFG-SERIAL-NUM   =  HV-MFG-SERIAL-NUM                 00079100
   AND  BEN-CUSTOMER-ID      =  HV-CUSTOMER-ID                    00079100
    EVALUATE   TRUE                                               00079100
     WHEN EIBAID = DFHPF4                                         00083400
     WHEN EIBAID = DFHPF16                                        00083400
       MOVE  WS-PFKEY-MSG2           TO  PFKEY-MSGO               00079100
       MOVE  WS-B102-NUM             TO  MSGNUMO
                                         STC-HELP-MSG             00063300
       MOVE  WS-B102-MSG             TO  MSG1O
       SET  STU-UPDATE       TO  TRUE
       SET  STU-PREV-KEY-PF4  TO  TRUE

     WHEN EIBAID = DFHPF9                                         00083400
     WHEN EIBAID = DFHPF21                                        00083400
       SET  WS-GET-PREVIOUS-ROW  TO  TRUE                         00014500
       PERFORM  F090-DISP-PREV-ROW THRU
                F090-DISP-PREV-ROW-EXIT
       SET  STU-PREV-KEY-PF9  TO  TRUE

     WHEN EIBAID = DFHPF10                                        00083400
     WHEN EIBAID = DFHPF22                                        00083400
       SET  WS-GET-NEXT-ROW  TO  TRUE                             00014400
       PERFORM  F100-DISP-NEXT-ROW THRU
                F100-DISP-NEXT-ROW-EXIT
       SET  STU-PREV-KEY-PF10 TO  TRUE

     WHEN OTHER                                                   00083400
       PERFORM  R050-SELECT-CURRENT-ROW THRU                      00013400
```

```
                R050-SELECT-CURRENT-ROW-EXIT                    00013400
            IF  WS-EDIT-ERROR
              GO TO  F000-LATERAL-PRE-EXIT
            END-IF
            MOVE  BENEFITS-ROW          TO  STU-BEN-ROW         00088800
            PERFORM  N000-FILL-SCREEN THRU                      00088800
                     N000-FILL-SCREEN-EXIT                      00088800

          END-EVALUATE                                          00079100
        ELSE                                                    00079100
          SET  WS-GET-CURRENT-ROW  TO  TRUE                     00014600
          MOVE  HV-SERIAL-NUM           TO  BEN-SERIAL-NUM
          MOVE  HV-MFG-SERIAL-NUM       TO  BEN-MFG-SERIAL-NUM
          MOVE  HV-CUSTOMER-ID          TO  BEN-CUSTOMER-ID
          PERFORM F100-DISP-NEXT-ROW THRU
                  F100-DISP-NEXT-ROW-EXIT
        END-IF                                                  00079100

       WHEN  STU-UPDATE                                         00079100
         MOVE  STU-BEN-ROW              TO  BENEFITS-ROW        00079100
         MOVE  WS-PFKEY-MSG2            TO  PFKEY-MSGO          00079100
         EVALUATE   TRUE                                        00079100
          WHEN EIBAID = DFHPF6                                  00083400
          WHEN EIBAID = DFHPF18                                 00083400
            MOVE  WS-PFKEY-MSG1         TO  PFKEY-MSGO          00079100
            MOVE  WS-B103-NUM           TO  MSGNUMO
                                            STC-HELP-MSG        00063300
            MOVE  WS-B103-MSG           TO  MSG1O
            SET  STU-BROWSE TO TRUE
            SET  STU-PREV-KEY-PF6  TO  TRUE

          WHEN EIBAID = DFHPF2                                  00083400
          WHEN EIBAID = DFHPF14                                 00083400
            MOVE  WS-PFKEY-MSG3         TO  PFKEY-MSGO          00079100
            MOVE  WS-B105-NUM           TO  MSGNUMO
                                            STC-HELP-MSG        00063300
            MOVE  WS-B105-MSG           TO  MSG1O
            SET  STU-PREV-KEY-PF2  TO  TRUE

          WHEN EIBAID = DFHPF5                                  00083400
          WHEN EIBAID = DFHPF22                                 00083400
            PERFORM  F500-COMMIT-PROC THRU
                     F500-COMMIT-PROC-EXIT
            SET  STU-PREV-KEY-PF5  TO  TRUE

          WHEN OTHER                                            00083400
            SET  STU-PREV-KEY-ENTER  TO  TRUE

         END-EVALUATE                                           00079100
       END-EVALUATE                                             00079100
       .                                                        00089000
F000-LATERAL-PRE-EXIT.                                          00089300

       EVALUATE   TRUE                                          00079100
        WHEN  STU-BROWSE                                        00079100
          PERFORM  F810-SET-ATTR-BROWSE THRU
                   F810-SET-ATTR-BROWSE-EXIT
```

```
                                                                        00079100
        EVALUATE  TRUE
         WHEN  STU-SEARCH-SER-NUM
          MOVE   -1                       TO  CARD-SER1L
          MOVE   'BROWSE: CARD SER#'      TO  MSG2O (1 : 18)
                                                                        00079100
         WHEN  STU-SEARCH-MFG-SER-NUM
          MOVE   -1                       TO  MFG-SER1L
          MOVE   'BROWSE: MFG SER#'       TO  MSG2O (1 : 18)
                                                                        00079100
         WHEN  STU-SEARCH-CUST-ID
          MOVE   -1                       TO  CUST-IDL
          MOVE   'BROWSE: CUST ID'        TO  MSG2O (1 : 18)
                                                                        00079100
         WHEN  OTHER
          MOVE   -1                       TO  CARD-SER1L
        END-EVALUATE
                                                                        00079100
       WHEN  STU-UPDATE                                                 00079100
        PERFORM  F800-SET-ATTR-UPDATE THRU
                 F800-SET-ATTR-UPDATE-EXIT
                                                                        00079100
        MOVE   -1                         TO  CARD-SER1L
                                                                        00079100
      END-EVALUATE                                                      00079100

      SET  STC-GOING-LATERAL          TO  TRUE                          00088800
      SET  CD-SCR-ROUTER-EXIT         TO  TRUE                          00088900
      .                                                                 00089000
  F000-LATERAL-PROCESS-EXIT.                                            00089300
      EXIT.                                                             00089400
                                                                        00136800
  F090-DISP-PREV-ROW.
*****************************************************************     00137000
*  SETS SPECIFIED FIELDS AS UNPROTECTED, ALLOWING UPDATE OF     *       00137100
*  THESE FIELDS.                                                *       00137200
*****************************************************************     00137300
                                                                        00137400
      MOVE 'F090PREV'                  TO WORK-PARA                     00137500
                                                                        00136800
      PERFORM R200-STARTBR-INDICES THRU
              R200-STARTBR-INDICES-EXIT

      IF  WS-EDIT-ERROR
        SET  STC-EDIT-ERROR           TO  TRUE                          00088800
        GO TO F090-DISP-PREV-ROW-EXIT
      END-IF

      PERFORM R350-READPREV-INDICES THRU
              R350-READPREV-INDICES-EXIT


      PERFORM R350-READPREV-INDICES THRU
              R350-READPREV-INDICES-EXIT

      IF  WS-END-SEARCH
        MOVE  WS-SAVE-MFB1-KEY         TO  WS-MFB1-KEY                  00014200
```

```
      END-IF                                                              00013400
      EVALUATE  TRUE                                                      00013400
       WHEN  STU-SEARCH-SER-NUM                                           00013400
        MOVE  WS-MFB1-X1-SERIAL-NUM       TO  BEN-SERIAL-NUM
        MOVE  WS-MFB1-X1-CUSTOMER-ID      TO  BEN-CUSTOMER-ID
        MOVE  WS-MFB1-X1-BENEFIT-TYPE     TO  BEN-BENEFIT-TYPE
        MOVE  WS-MFB1-X1-EFFECTIVE-DATE   TO  WS-DATE-TIME
        PERFORM Q380-X14-TO-ORA THRU                                      00079100
                Q380-X14-TO-ORA-EXIT                                      00079100
        MOVE  WS-ORA-DT-TM                TO  BEN-EFFECTIVE-DATE
        MOVE  WS-MFB1-X1-MFG-SERIAL-NUM   TO  BEN-MFG-SERIAL-NUM
                                                                          00013400
       WHEN  STU-SEARCH-MFG-SER-NUM                                       00013400
        MOVE  WS-MFB1-X2-SERIAL-NUM       TO  BEN-SERIAL-NUM
        MOVE  WS-MFB1-X2-CUSTOMER-ID      TO  BEN-CUSTOMER-ID
        MOVE  WS-MFB1-X2-BENEFIT-TYPE     TO  BEN-BENEFIT-TYPE
        MOVE  WS-MFB1-X2-EFFECTIVE-DATE   TO  WS-DATE-TIME
        PERFORM Q380-X14-TO-ORA THRU                                      00079100
                Q380-X14-TO-ORA-EXIT                                      00079100
        MOVE  WS-ORA-DT-TM                TO  BEN-EFFECTIVE-DATE
        MOVE  WS-MFB1-X2-MFG-SERIAL-NUM   TO  BEN-MFG-SERIAL-NUM
                                                                          00013400
       WHEN  STU-SEARCH-CUST-ID                                           00013400
        MOVE  WS-MFB1-X3-SERIAL-NUM       TO  BEN-SERIAL-NUM
        MOVE  WS-MFB1-X3-CUSTOMER-ID      TO  BEN-CUSTOMER-ID
        MOVE  WS-MFB1-X3-BENEFIT-TYPE     TO  BEN-BENEFIT-TYPE
        MOVE  WS-MFB1-X3-EFFECTIVE-DATE   TO  WS-DATE-TIME
        PERFORM Q380-X14-TO-ORA THRU                                      00079100
                Q380-X14-TO-ORA-EXIT                                      00079100
        MOVE  WS-ORA-DT-TM                TO  BEN-EFFECTIVE-DATE
        MOVE  WS-MFB1-X3-MFG-SERIAL-NUM   TO  BEN-MFG-SERIAL-NUM
                                                                          00013400
      END-EVALUATE                                                        00013400
                                                                          00013400
      PERFORM  R050-SELECT-CURRENT-ROW THRU                               00013400
               R050-SELECT-CURRENT-ROW-EXIT                               00013400

      IF  WS-EDIT-OK
        MOVE  BENEFITS-ROW              TO  STU-BEN-ROW                   00088800

        PERFORM  N000-FILL-SCREEN THRU                                    00088800
                 N000-FILL-SCREEN-EXIT                                    00088800
      ELSE
        SET  STC-EDIT-ERROR          TO  TRUE                             00088800
      END-IF
      .                                                                   00089000
  F090-DISP-PREV-ROW-EXIT.
      EXIT.                                                               00089400
                                                                          00136800
  F100-DISP-NEXT-ROW.
 ************************************************************************ 00137000
 *  DETERMINES NEXT ROW TO BE FETCHED, FETCHES THAT ROW AND      *        00137100
 *  DISPLAYS ITS COLUMNS.                                        *        00137200
 ************************************************************************ 00137300
                                                                          00137400
```

```
      MOVE 'F100     ' TO WORK-PARA                                    00137500
                                                                       00136800
      PERFORM R200-STARTBR-INDICES THRU
              R200-STARTBR-INDICES-EXIT

      IF  WS-EDIT-ERROR
        SET  STC-EDIT-ERROR          TO  TRUE                          00088800
        GO TO F100-DISP-NEXT-ROW-EXIT
      END-IF

      PERFORM R300-READNEXT-INDICES THRU
              R300-READNEXT-INDICES-EXIT

*     MOVE WS-MFB1-KEY             TO    MSG3O
      IF  WS-EDIT-ERROR
        SET  STC-EDIT-ERROR          TO  TRUE                          00088800
        GO TO F100-DISP-NEXT-ROW-EXIT
      END-IF

      EVALUATE  TRUE
       WHEN  STU-INITIAL-SCREEN
        AND  WS-END-SEARCH
         PERFORM R350-READPREV-INDICES THRU
                 R350-READPREV-INDICES-EXIT
                                                                       00137500
         PERFORM R350-READPREV-INDICES THRU
                 R350-READPREV-INDICES-EXIT
                                                                       00137500
       WHEN   WS-GET-CURRENT-ROW                                       00014600
        AND   WS-END-SEARCH
         PERFORM R350-READPREV-INDICES THRU
                 R350-READPREV-INDICES-EXIT

         PERFORM R350-READPREV-INDICES THRU
                 R350-READPREV-INDICES-EXIT

       WHEN   WS-GET-NEXT-ROW
        IF    HV-SERIAL-NUM      =  BEN-SERIAL-NUM
         AND  HV-MFG-SERIAL-NUM  =  BEN-MFG-SERIAL-NUM
         AND  HV-CUSTOMER-ID     =  BEN-CUSTOMER-ID
          PERFORM R300-READNEXT-INDICES THRU
                  R300-READNEXT-INDICES-EXIT
        END-IF
        IF  WS-END-SEARCH
          MOVE  WS-SAVE-MFB1-KEY        TO  WS-MFB1-KEY              00014200
        END-IF
      END-EVALUATE
                                                                       00013400
      IF  WS-EDIT-ERROR
        SET  STC-EDIT-ERROR          TO  TRUE                          00088800
        GO TO F100-DISP-NEXT-ROW-EXIT
      END-IF
                                                                       00013400
      EVALUATE  TRUE
       WHEN  STU-SEARCH-SER-NUM                                        00013400
        MOVE  WS-MFB1-X1-SERIAL-NUM      TO  BEN-SERIAL-NUM
        MOVE  WS-MFB1-X1-CUSTOMER-ID     TO  BEN-CUSTOMER-ID
```

```
        MOVE  WS-MFB1-X1-BENEFIT-TYPE    TO  BEN-BENEFIT-TYPE
        MOVE  WS-MFB1-X1-EFFECTIVE-DATE  TO  WS-DATE-TIME
        PERFORM Q380-X14-TO-ORA THRU                                    00079100
                Q380-X14-TO-ORA-EXIT                                    00079100
        MOVE  WS-ORA-DT-TM               TO  BEN-EFFECTIVE-DATE
        MOVE  WS-MFB1-X1-MFG-SERIAL-NUM  TO  BEN-MFG-SERIAL-NUM
                                                                        00013400
       WHEN  STU-SEARCH-MFG-SER-NUM                                     00013400
        MOVE  WS-MFB1-X2-SERIAL-NUM      TO  BEN-SERIAL-NUM
        MOVE  WS-MFB1-X2-CUSTOMER-ID     TO  BEN-CUSTOMER-ID
        MOVE  WS-MFB1-X2-BENEFIT-TYPE    TO  BEN-BENEFIT-TYPE
        MOVE  WS-MFB1-X2-EFFECTIVE-DATE  TO  WS-DATE-TIME
        PERFORM Q380-X14-TO-ORA THRU                                    00079100
                Q380-X14-TO-ORA-EXIT                                    00079100
        MOVE  WS-ORA-DT-TM               TO  BEN-EFFECTIVE-DATE
        MOVE  WS-MFB1-X2-MFG-SERIAL-NUM  TO  BEN-MFG-SERIAL-NUM
                                                                        00013400
       WHEN  STU-SEARCH-CUST-ID                                         00013400
        MOVE  WS-MFB1-X3-SERIAL-NUM      TO  BEN-SERIAL-NUM
        MOVE  WS-MFB1-X3-CUSTOMER-ID     TO  BEN-CUSTOMER-ID
        MOVE  WS-MFB1-X3-BENEFIT-TYPE    TO  BEN-BENEFIT-TYPE
        MOVE  WS-MFB1-X3-EFFECTIVE-DATE  TO  WS-DATE-TIME
        PERFORM Q380-X14-TO-ORA THRU                                    00079100
                Q380-X14-TO-ORA-EXIT                                    00079100
        MOVE  WS-ORA-DT-TM               TO  BEN-EFFECTIVE-DATE
        MOVE  WS-MFB1-X3-MFG-SERIAL-NUM  TO  BEN-MFG-SERIAL-NUM
                                                                        00013400
    END-EVALUATE
                                                                        00013400
    PERFORM  R050-SELECT-CURRENT-ROW THRU                               00013400
             R050-SELECT-CURRENT-ROW-EXIT                               00013400

    IF  WS-EDIT-OK
      MOVE  BENEFITS-ROW             TO  STU-BEN-ROW                  00088800

      PERFORM  N000-FILL-SCREEN THRU                                    00088800
               N000-FILL-SCREEN-EXIT                                    00088800
    ELSE
      SET  STC-EDIT-ERROR          TO  TRUE                           00088800
    END-IF
    .                                                                   00089000
 F100-DISP-NEXT-ROW-EXIT.
    EXIT.                                                               00089400
                                                                        00136800
 F200-TOGGLE-HOLD.
*****************************************************************     00137000
*  TOGGLES (SETS/RESETS) BEN-HOLD-CDE SWITCH.                   *      00137100
*****************************************************************     00137300
                                                                        00137400
    MOVE 'F200    ' TO WORK-PARA                                        00137500
                                                                        00137400
    EVALUATE   TRUE                                                     00137500
     WHEN  BEN-HOLD-THIS-BENEFIT                                        00013400
      PERFORM  R100-SEL-UPD-CURR-ROW THRU                               00013400
               R100-SEL-UPD-CURR-ROW-EXIT                               00013400
      IF  ORA-SQL-SUCCESSFUL                                            00013400
        SET  BEN-NO-EXTERNAL-HOLD  TO  TRUE                             00013400
```

```
    PERFORM  R140-UPD-HOLD-COLS THRU                              00013400
             R140-UPD-HOLD-COLS-EXIT                              00013400
    PERFORM  R150-UPDATE-CURR-ROW THRU                            00013400
             R150-UPDATE-CURR-ROW-EXIT                            00013400
    IF  ORA-SQL-SUCCESSFUL                                        00013400
      EXEC  SQL                                                   00013400
         COMMIT  WORK                                             00013400
      END-EXEC                                                    00013400
      PERFORM  R145-UPD-HOLD-SCRN-DATA THRU
               R145-UPD-HOLD-SCRN-DATA-EXIT
    END-IF                                                        00013400
  ELSE                                                            00013400
    SET  WS-EDIT-ERROR                                            00063400
         STC-EDIT-ERROR     TO  TRUE                              00063400
                                                                  00063100
    MOVE WS-B188-NUM              TO  MSGNUMO                     00063200
                                      STC-HELP-MSG                00063300
    MOVE WS-B188-MSG              TO  MSG1O                       00063400
  END-IF                                                          00013400
                                                                  00013400
 WHEN  BEN-NO-EXTERNAL-HOLD                                       00013400
  PERFORM  R100-SEL-UPD-CURR-ROW THRU                             00013400
           R100-SEL-UPD-CURR-ROW-EXIT                             00013400
  IF  ORA-SQL-SUCCESSFUL                                          00013400
    SET  BEN-HOLD-THIS-BENEFIT  TO  TRUE                          00013400
    PERFORM  R140-UPD-HOLD-COLS THRU                              00013400
             R140-UPD-HOLD-COLS-EXIT                              00013400
    PERFORM  R150-UPDATE-CURR-ROW THRU                            00013400
             R150-UPDATE-CURR-ROW-EXIT                            00013400
    IF  ORA-SQL-SUCCESSFUL                                        00013400
      EXEC  SQL                                                   00013400
         COMMIT  WORK                                             00013400
      END-EXEC                                                    00013400
      PERFORM  R145-UPD-HOLD-SCRN-DATA THRU
               R145-UPD-HOLD-SCRN-DATA-EXIT
    END-IF                                                        00013400
  ELSE                                                            00013400
    SET  WS-EDIT-ERROR                                            00063400
         STC-EDIT-ERROR     TO  TRUE                              00063400
                                                                  00063100
    MOVE WS-B188-NUM              TO  MSGNUMO                     00063200
                                      STC-HELP-MSG                00063300
    MOVE WS-B188-MSG              TO  MSG1O                       00063400
  END-IF                                                          00013400
                                                                  00013400
 WHEN  OTHER                                                      00013400
  MOVE  '**NOT DEFINED *'       TO  HOLD-DESCO                    00013400
                                                                  00137400
  SET  WS-EDIT-ERROR                                              00063400
       STC-EDIT-ERROR     TO  TRUE                                00063400
                                                                  00063100
  MOVE WS-B187-NUM              TO  MSGNUMO                     00063200
                                    STC-HELP-MSG                  00063300
  MOVE WS-B187-MSG              TO  MSG1O                       00063400
  MOVE  BEN-HOLD-CDE            TO  MSG1O (27 : 1)
                                                                  00013400
END-EVALUATE                                                      00013400
```

```
     .                                                                  00089000
 F200-TOGGLE-HOLD-EXIT.
     EXIT.                                                              00089400
                                                                        00136800
 F500-COMMIT-PROC.
*****************************************************************     00137000
*  PROCESSES THE COMMIT OF ANY PREVIOUS UPDATES                 *      00137100
*****************************************************************     00137300
                                                                        00137400
     MOVE 'F500    ' TO WORK-PARA                                     00137500
                                                                        00137400
     EVALUATE  TRUE                                                   00137500
      WHEN  STU-PREV-KEY-PF2
        PERFORM F200-TOGGLE-HOLD THRU
                F200-TOGGLE-HOLD-EXIT
                                                                        00137400
     END-EVALUATE                                                     00137500
     .                                                                  00089000
 F500-COMMIT-PROC-EXIT.
     EXIT.                                                              00089400
/                                                                       00136800
 F800-SET-ATTR-UPDATE.
*****************************************************************00149600
*****  SETS PROTECT ATTRIBUTE ON BROWSE KEY FIELDS      **********00149700
*****************************************************************00149900
                                                                        00137400
     MOVE 'F800    ' TO WORK-PARA                                     00137500
                                                                        00150100
     MOVE  X'F9'        TO                                            00150200
       CARD-SER1A  CARD-SER2A  CUST-IDA   MFG-SER1A  MFG-SER2A        00150300
     .                                                                  00089000
 F800-SET-ATTR-UPDATE-EXIT.
     EXIT.                                                              00089400
                                                                        00136800
 F810-SET-ATTR-BROWSE.
*****************************************************************     00137000
*  UNPROTECTS THE BROWSE FIELDS                                 *      00137100
*****************************************************************     00137300
                                                                        00137400
     MOVE 'F810    '                    TO  WORK-PARA                 00137500
                                                                        00137400
     MOVE  DFHVAL        TO                                            00137500
*      CARD-SER1A  CARD-SER2A  CUST-IDA   MFG-SER1A  MFG-SER2A        00150300
       CARD-SER1A  CARD-SER2A             MFG-SER1A  MFG-SER2A        00150300
     .                                                                  00089000
 F810-SET-ATTR-BROWSE-EXIT.
     EXIT.                                                              00089400
/                                                                       00136800
 M200-EDIT-SEARCH-KEY.                                                  00079100
*****************************************************************     00137000
*  EDITS THE FIELDS USED IN THE SEARCH KEY                      *      00137100
*  DETERMINES IF ANY OF THE FIELDS HAVE BEEN CHANGED            *      00137100
*  A CHANGED FIELD DENOTES THE OPERATOR'S DESIRE TO SEARCH BY   *      00137100
*  THAT PARTICULAR FIELD.                                       *      00137100
*****************************************************************     00137300
                                                                        00137400
     MOVE 'M200' TO WORK-PARA                                           00137500
```

```
                                                                        00137400
     PERFORM  M201-EDIT-SER-NUM THRU                                    00079100
              M201-EDIT-SER-NUM-EXIT                                    00079100
                                                                        00137400
     PERFORM  M202-EDIT-MFG-SER-NUM THRU                                00079100
              M202-EDIT-MFG-SER-NUM-EXIT                                00079100
                                                                        00137400
     PERFORM  M203-EDIT-CUST-ID THRU                                    00079100
              M203-EDIT-CUST-ID-EXIT                                    00079100
                                                                        00137400
     IF  WS-EDIT-OK                                                     00079100
       EVALUATE  TRUE                                                   00079100
        WHEN  BEN-SERIAL-NUM      NOT =  HV-SERIAL-NUM                  00079100
         SET  STU-SEARCH-SER-NUM  TO  TRUE                              00079100
         SET  HV-FIRST-MFG-SER-NUM  TO  TRUE                            00079100
         SET  HV-FIRST-CUST-ID  TO  TRUE                                00079100
                                                                        00137400
        WHEN  BEN-MFG-SERIAL-NUM  NOT =  HV-MFG-SERIAL-NUM              00079100
         SET  STU-SEARCH-MFG-SER-NUM  TO  TRUE                          00079100
         SET  HV-FIRST-SER-NUM  TO  TRUE                                00079100
         SET  HV-FIRST-CUST-ID  TO  TRUE                                00079100
                                                                        00137400
        WHEN  BEN-CUSTOMER-ID     NOT =  HV-CUSTOMER-ID                 00079100
         SET  STU-SEARCH-CUST-ID  TO  TRUE                              00079100
         SET  HV-FIRST-SER-NUM  TO  TRUE                                00079100
         SET  HV-FIRST-MFG-SER-NUM  TO  TRUE                            00079100
       END-EVALUATE                                                     00079100
     END-IF                                                             00079100
     .                                                                  00089000
 M200-EDIT-SEARCH-KEY-EXIT.                                             00079100
     EXIT.                                                              00089400
/                                                                       00136800
 M201-EDIT-SER-NUM.                                                     00079100
*****************************************************************     00137000
*  EDITS THE SERIAL NUMBER ENTERED ON THE SCREEN                *     00137100
*****************************************************************     00137300
                                                                        00137400
     MOVE 'M201' TO WORK-PARA                                           00137500
                                                                        00137400
     MOVE  CARD-SER1I              TO  NC-INPUT-NUMBER-VAR              00275300
     MOVE     5                    TO  NC-NUM-SIZE                      00275400
     SET  NC-INTEGER-CHECK  TO  TRUE                                    00275500
     PERFORM  NUMCHK-EDIT-RTN THRU                                      00275600
              NUMCHK-EDIT-RTN-EXIT                                      00275700
     IF  NOT NC-NUMERIC-OK                                              00275800
       IF  WS-EDIT-OK                                                   00275900
         MOVE  WS-B121-NUM          TO    MSGNUMO                     00276000
                                          STC-HELP-MSG                  00276100
         MOVE WS-B121-MSG           TO    MSG1O                       00276300
         MOVE  'SER NUM DEV #'      TO    MSG1O (24 : 15)             00276300
         MOVE -1                    TO    CARD-SER1L                  00276500
       END-IF                                                           00276600
       MOVE DFHUNIMD                TO    CARD-SER1A                  00276700
       SET  WS-EDIT-ERROR  TO    TRUE                                   00276800
     END-IF                                                             00275800
     IF  WS-EDIT-OK                                                     00275900
       MOVE  NC-NUMBER-MASK-VAR     TO  WS-OSN-HI-5                    00275900
```

```
     END-IF                                                              00275900
                                                                         00137400
     MOVE  CARD-SER2I              TO  NC-INPUT-NUMBER-VAR               00275300
     MOVE     10                   TO  NC-NUM-SIZE                       00275400
     SET  NC-INTEGER-CHECK  TO  TRUE                                     00275500
     PERFORM  NUMCHK-EDIT-RTN THRU                                       00275600
              NUMCHK-EDIT-RTN-EXIT                                       00275700
     IF  NOT NC-NUMERIC-OK                                               00275800
       IF  WS-EDIT-OK                                                    00275900
         MOVE  WS-B121-NUM           TO    MSGNUMO                     00276000
                                           STC-HELP-MSG                  00276100
         MOVE WS-B121-MSG            TO    MSG1O                       00276300
         MOVE  'SER NUM GMT  '       TO    MSG1O (24 : 15)             00276300
         MOVE -1                     TO    CARD-SER2L                  00276500
       END-IF                                                            00276600
       MOVE DFHUNIMD                 TO    CARD-SER2A                  00276700
       SET  WS-EDIT-ERROR  TO    TRUE                                    00276800
     END-IF                                                              00275800
     IF  WS-EDIT-OK                                                      00275900
       MOVE  NC-NUMBER-MASK-VAR      TO  WS-OSN-LO-10                    00275900
       MOVE  WS-SERIAL-NUM-X15       TO  HV-SERIAL-NUM                   00275900
     END-IF                                                              00275900
     .                                                                   00089000
 M201-EDIT-SER-NUM-EXIT.                                                 00079100
     EXIT.                                                               00089400
/                                                                        00136800
 M202-EDIT-MFG-SER-NUM.                                                  00079100
*****************************************************************      00137000
*  EDITS THE MANUFACTURER'S SERIAL NUMBER                       *      00137100
*****************************************************************      00137300
                                                                         00137400
     MOVE 'M202' TO WORK-PARA                                            00137500
                                                                         00137400
     MOVE  MFG-SER1I               TO  NC-INPUT-NUMBER-VAR               00275300
     MOVE     3                    TO  NC-NUM-SIZE                       00275400
     SET  NC-INTEGER-CHECK  TO  TRUE                                     00275500
     PERFORM  NUMCHK-EDIT-RTN THRU                                       00275600
              NUMCHK-EDIT-RTN-EXIT                                       00275700
     IF  NOT NC-NUMERIC-OK                                               00275800
       IF  WS-EDIT-OK                                                    00275900
         MOVE  WS-B121-NUM           TO    MSGNUMO                     00276000
                                           STC-HELP-MSG                  00276100
         MOVE WS-B121-MSG            TO    MSG1O                       00276300
         MOVE  'MFG NUM #      '     TO    MSG1O (24 : 15)             00276300
         MOVE -1                     TO    MFG-SER1L                   00276500
       END-IF                                                            00276600
       MOVE DFHUNIMD                 TO    MFG-SER1A                   00276700
       SET  WS-EDIT-ERROR  TO    TRUE                                    00276800
     END-IF                                                              00275800
     IF  WS-EDIT-OK                                                      00275900
       MOVE  NC-NUMBER-MASK-VAR      TO  WS-MSN-HI-3                     00275900
     END-IF                                                              00275900
                                                                         00137400
     MOVE  MFG-SER2I               TO  NC-INPUT-NUMBER-VAR               00275300
     MOVE     8                   TO  NC-NUM-SIZE                       00275400
     SET  NC-INTEGER-CHECK  TO  TRUE                                     00275500
     PERFORM  NUMCHK-EDIT-RTN THRU                                       00275600
```

```
            NUMCHK-EDIT-RTN-EXIT                                        00275700
     IF  NOT NC-NUMERIC-OK                                              00275800
       IF  WS-EDIT-OK                                                   00275900
         MOVE  WS-B121-NUM          TO    MSGNUMO                     00276000
                                          STC-HELP-MSG                00276100
         MOVE WS-B121-MSG           TO    MSG1O                       00276300
         MOVE  'MFG SERIAL#  '      TO    MSG1O (24 : 15)             00276300
         MOVE -1                    TO    MFG-SER2L                   00276500
       END-IF                                                           00276600
       MOVE DFHUNIMD                TO    MFG-SER2A                   00276700
       SET  WS-EDIT-ERROR  TO    TRUE                                  00276800
     END-IF                                                             00275800
     IF  WS-EDIT-OK                                                     00275900
       MOVE  NC-NUMBER-MASK-VAR     TO  WS-MSN-LO-8                   00275900
       MOVE  WS-MFG-SERIAL-NUM-X11  TO  HV-MFG-SERIAL-NUM             00275900
     END-IF                                                             00275900
     .                                                                  00089000
 M202-EDIT-MFG-SER-NUM-EXIT.                                            00079100
     EXIT.                                                              00089400
                                                                        00136800
 M203-EDIT-CUST-ID.                                                     00079100
*****************************************************************     00137000
*  EDITS THE CUSTOMER ID                                        *     00137100
*****************************************************************     00137300
                                                                        00137400
     MOVE 'M203' TO WORK-PARA                                           00137500
                                                                        00137400
     MOVE  CUST-IDI                   TO  WS-CUSTOMER-ID-X14         00275900
     PERFORM  VARYING  I  FROM 1  BY  1                                 00275900
          UNTIL  I  >  14                                               00275900
       IF  WS-CI-CHAR (I)  =  SPACE                                     00275900
         IF  WS-EDIT-OK                                                 00275900
           MOVE  WS-B121-NUM          TO    MSGNUMO                   00276000
                                            STC-HELP-MSG              00276100
           MOVE WS-B121-MSG           TO    MSG1O                     00276300
           MOVE  'CUSTOMER ID  '      TO    MSG1O (24 : 15)           00276300
           MOVE -1                    TO    CUST-IDL                  00276500
         END-IF                                                         00276600
         MOVE DFHUNIMD                TO    CUST-IDA                  00276700
         SET  WS-EDIT-ERROR  TO    TRUE                               00276800
         MOVE  15                     TO  I                           00276800
       END-IF                                                           00275800
     END-PERFORM                                                        00275900
                                                                        00137400
     IF  WS-EDIT-OK                                                     00275900
       MOVE  WS-CUSTOMER-ID-X14       TO  HV-CUSTOMER-ID              00275900
     END-IF                                                             00275900
     .                                                                  00089000
 M203-EDIT-CUST-ID-EXIT.                                                00079100
     EXIT.                                                              00089400
/                                                                       00136800
 N000-FILL-SCREEN.                                                      00136900
*****************************************************************     00137000
*  THIS PARAGRAPH CONTROLS ALL PROCESSING RELATED TO DATA MOVE- *     00137100
*  MENT TO THE SCREEN AREA.                                     *     00137200
*****************************************************************     00137300
                                                                        00137400
```

```
MOVE 'N000' TO WORK-PARA                                          00137500
                                                                  00137400
MOVE  BEN-SN-DEV-NUM           TO  WS-DISP-NUM-5
MOVE  WS-DISP-NUM-5            TO  CARD-SER1O
MOVE  BEN-SN-GMT               TO  WS-DISP-NUM-10
MOVE  WS-DISP-NUM-10           TO  CARD-SER2O
                                                                  00137400
MOVE  BEN-CUSTOMER-ID          TO  CUST-IDO
                                                                  00137400
MOVE  BEN-EFFECTIVE-DATE       TO  WS-ORA-DT-TM
PERFORM  Q300-GET-DISP-DATE THRU
         Q300-GET-DISP-DATE-EXIT
MOVE  WS-DATE-DISP-10          TO  EFF-DATEO
MOVE  WS-TIME-DISP-8           TO  EFF-TIMEO
                                                                  00137400
MOVE  BEN-BENEFIT-TYPE         TO  BEN-TYPEO
EVALUATE  TRUE                                                    00013400
 WHEN  BEN-METRO-CHECK                                            00013400
  MOVE  ' METRO CHECK   '      TO  BEN-DESCO                      00013400
                                                                  00013400
 WHEN  OTHER                                                      00013400
  MOVE  '  NOT DEFINED  '      TO  BEN-DESCO                      00013400
END-EVALUATE                                                      00013400
MOVE  '* NOT DEFINED *'        TO  BND-BENEFIT-DESC               00013400
EXEC  SQL  SELECT                                                 00013400
     BENEFIT_DESC                                                 00013400
   INTO                                                           00013400
     :BND-BENEFIT-DESC                                            00013400
 FROM  MCHECK.BENEFITS_DEFINITION                                 00013400
   WHERE                                                          00013400
     BENEFIT_TYPE  =  RPAD(:BEN-BENEFIT-TYPE,5,' ')               00013400
END-EXEC                                                          00013400
MOVE  BND-BENEFIT-DESC-ARR     TO  BEN-DESCO                      00013400

MOVE  BEN-MFG-SERIAL-NUM (1 : 3) TO  MFG-SER1O
MOVE  BEN-MFG-SERIAL-NUM (4 : 8) TO  MFG-SER2O

MOVE     ZERO                  TO  WS-BIN-2N
MOVE  BEN-LAST-AUTH-SEQ-NUM    TO  WS-B2-LO
MOVE  WS-BIN-2N                TO  WS-COMP-DISP-4
MOVE  WS-COMP-DISP-4           TO  AUTH-CDEO

MOVE  BEN-LOAD-DT-TM           TO  WS-ORA-DT-TM
PERFORM  Q300-GET-DISP-DATE THRU
         Q300-GET-DISP-DATE-EXIT
MOVE  WS-DATE-DISP-10          TO  LOAD-DATEO
MOVE  WS-TIME-DISP-8           TO  LOAD-TIMEO

MOVE  BEN-EXPIRATION-DATE      TO  WS-ORA-DT-TM
PERFORM  Q300-GET-DISP-DATE THRU
         Q300-GET-DISP-DATE-EXIT
MOVE  WS-DATE-DISP-10          TO  EXP-DATEO
MOVE  WS-TIME-DISP-8           TO  EXP-TIMEO
                                                                  00137400
MOVE  BEN-INITIAL-VAL-AMT      TO  WS-DISP-DEC-FMT-5              00013400
MOVE  WS-DISP-DEC-FMT-5        TO  INIT-AMTO                      00013400
                                                                  00137400
```

```
MOVE  BEN-REM-VAL-AMT          TO  WS-DISP-DEC-FMT-5          00013400
MOVE  WS-DISP-DEC-FMT-5        TO  REM-AMTO                   00013400
                                                              00013400
MOVE  BEN-LAST-CLAIM-HOLD-CDE  TO  EVHLDCDEO                  00013400
EVALUATE  TRUE                                                00013400
 WHEN  BEN-ON-HOLD-PREV-EUB2                                  00013400
  MOVE  '==> ON HOLD    '      TO  EVHLDDESCO                 00013400
                                                              00013400
 WHEN  BEN-AVAILABLE                                          00013400
  MOVE  '  AVAILABLE    '      TO  EVHLDDESCO                 00013400
                                                              00013400
 WHEN  OTHER                                                  00013400
  MOVE  '**UNKNOWN CODE*'      TO  EVHLDDESCO                 00013400
                                                              00013400
END-EVALUATE                                                  00013400
                                                              00137400
MOVE  BEN-LAST-CLAIM-VAL-AMT   TO  WS-DISP-DEC-FMT-5          00013400
MOVE  WS-DISP-DEC-FMT-5        TO  LCLM-AMTO                  00013400
                                                              00137400
MOVE  BEN-LAST-CLAIM-DT-TM     TO  WS-ORA-DT-TM
PERFORM  Q300-GET-DISP-DATE THRU
         Q300-GET-DISP-DATE-EXIT
MOVE  WS-DATE-DISP-10          TO  LCLM-DATEO
MOVE  WS-TIME-DISP-8           TO  LCLM-TIMEO
                                                              00013400
MOVE  BEN-LAST-AUTH-CDE        TO  AUTH-CDEO
                                                              00137400
MOVE BEN-LAST-RETR-REF-NUM     TO  RETREFNUMO
                                                              00137400
MOVE  BEN-LAST-REQUEST-TYPE    TO  LREQ-TYPEO
EVALUATE  TRUE                                                00013400
 WHEN  BEN-BENEFIT-LOADED                                     00013400
  MOVE  'INITIAL LOAD   '      TO  LREQ-DESCO                 00013400
                                                              00013400
 WHEN  BEN-BENEFIT-REQUEST                                    00013400
  MOVE  'EUB1 BEN REQST '      TO  LREQ-DESCO                 00013400
                                                              00013400
 WHEN  BEN-CLAIM-CONFIRM                                      00013400
  MOVE  'EUB5 CLM CONFRM'      TO  LREQ-DESCO                 00013400
                                                              00013400
 WHEN  BEN-HOLD-ALL-CUST-BENEFITS                             00013400
  MOVE  'EUB5 CLM CREDIT'      TO  LREQ-DESCO                 00013400
                                                              00013400
 WHEN  OTHER                                                  00013400
  MOVE  '  NOT DEFINED  '      TO  LREQ-DESCO                 00013400
END-EVALUATE                                                  00013400
                                                              00137400
MOVE  BEN-LAST-REQUEST-DT-TM   TO  WS-ORA-DT-TM
PERFORM  Q300-GET-DISP-DATE THRU
         Q300-GET-DISP-DATE-EXIT
MOVE  WS-DATE-DISP-10          TO  LREQ-DATEO
MOVE  WS-TIME-DISP-8           TO  LREQ-TIMEO
                                                              00013400
MOVE  BEN-HOLD-CDE             TO  HOLD-STO                   00013400
EVALUATE  TRUE                                                00013400
 WHEN  BEN-NO-EXTERNAL-HOLD                                   00013400
  MOVE  '  AVAILABLE    '      TO  HOLD-DESCO                 00013400
```

```
                                                                        00013400
      WHEN  BEN-HOLD-THIS-BENEFIT                                       00013400
       MOVE  'HOLD - SER #   '      TO  HOLD-DESCO                     00013400
                                                                        00013400
      WHEN  BEN-HOLD-ALL-CUST-BENEFITS                                  00013400
       MOVE  'HOLD - CUSTOMER'      TO  HOLD-DESCO                     00013400
                                                                        00013400
      WHEN  OTHER                                                       00013400
       MOVE  '  NOT DEFINED  '      TO  HOLD-DESCO                     00013400
     END-EVALUATE                                                       00013400
                                                                        00013400
     MOVE  BEN-HOLD-DT-TM           TO  WS-ORA-DT-TM
     PERFORM  Q300-GET-DISP-DATE THRU
              Q300-GET-DISP-DATE-EXIT
     MOVE  WS-DATE-DISP-10          TO  HOLD-DATEO
     MOVE  WS-TIME-DISP-8           TO  HOLD-TIMEO
                                                                        00137400
     MOVE  BEN-HOLD-USER-ID         TO  HOLD-UIDO
                                                                        00013400
     MOVE  BEN-UPDATE-DT-TM         TO  WS-ORA-DT-TM
     PERFORM  Q300-GET-DISP-DATE THRU
              Q300-GET-DISP-DATE-EXIT
     MOVE  WS-DATE-DISP-10          TO  UPD-DATEO
     MOVE  WS-TIME-DISP-8           TO  UPD-TIMEO
                                                                        00013400
     MOVE BEN-UPDATE-ACTION-CDE     TO  UPD-CDEO
     EVALUATE  TRUE                                                     00013400
      WHEN  BEN-NO-UPDATES                                              00013400
       MOVE  'INITIAL STATUS '      TO  UPD-DESCO                      00013400
                                                                        00013400
      WHEN  BEN-HOLD-THIS-BENEFIT                                       00013400
       MOVE  'UPD BEN TYPE   '      TO  UPD-DESCO                      00013400
                                                                        00013400
      WHEN  BEN-HOLD-ALL-CUST-BENEFITS                                  00013400
       MOVE  'UPD INITIAL VAL'      TO  UPD-DESCO                      00013400
                                                                        00013400
      WHEN  OTHER                                                       00013400
       MOVE  '  NOT DEFINED  '      TO  UPD-DESCO                      00013400
     END-EVALUATE                                                       00013400
                                                                        00137400
     MOVE  BEN-UPDATE-USER-ID       TO  UPD-UIDO
                                                                        00137400
     MOVE       ZERO                TO  WS-BIN-2N
     MOVE  BEN-LAST-AUTH-SEQ-NUM    TO  WS-B2-LO
     MOVE  WS-BIN-2N                TO  WS-DISP-ZZ9
     MOVE  WS-DISP-ZZ9 (3 : 1)      TO  ASEQ-NUMO
     .                                                                  00141200
 N000-FILL-SCREEN-EXIT.                                                 00141300
     EXIT.                                                              00141400

/                                                                       00136800
 Q000-SQLCODE-PROCESS.                                                  00136900
***************************************************************       00137000
*  INTERPETS ERROR AND DISPLAYS ON SCREEN                     *        00137100
***************************************************************       00137300
                                                                        00137400
*    MOVE 'Q000' TO WORK-PARA                                           00137500
```

```
                                                                        00137400
     MOVE  SPACES                        TO  SQLERRMC                   00073500
     MOVE SQLCODE                        TO ORA-NAMED-SQLCODE           00073500
                                            ORA-SQLCODE-DISP-4          00073600
     EVALUATE  TRUE
      WHEN ORA-SQL-SUCCESSFUL                                           00073800
       CONTINUE                                                         00063400
                                                                        00063400
      WHEN ORA-SQL-ROW-NOT-FOUND                                        00073900
       PERFORM  Q030-NOT-FOUND THRU                                     00063400
                Q030-NOT-FOUND-EXIT                                     00063400
                                                                        00063400
      WHEN  OTHER
       PERFORM  Q020-SQL-ERROR THRU                                     00063400
                Q020-SQL-ERROR-EXIT                                     00063400
                                                                        00063400
     END-EVALUATE
     .
 Q000-SQLCODE-PROCESS-EXIT.                                             00136900
     EXIT.

/                                                                       00136800
 Q020-SQL-ERROR.                                                        00136900
*************************************************************************00137000
*  INTERPETS ERROR AND DISPLAYS ON SCREEN                             * 00137100
*************************************************************************00137300
                                                                        00137400
     SET  WS-EDIT-ERROR                                                 00063400
          STC-EDIT-ERROR     TO  TRUE                                   00063400
                                                                        00063100
     MOVE WS-B181-NUM              TO  MSGNUMO                        00063200
                                       STC-HELP-MSG                     00063300
     MOVE WS-B181-MSG              TO  MSG1O                          00063400
     MOVE  SQLCODE                 TO  WS-SQLCODE-DISP
     MOVE  WS-SQLCODE-DISP         TO  MSG1O (11 : 5)
     MOVE  WORK-PARA               TO  MSG1O (23 : 8)
     MOVE  SQLERRMC                TO  MSG3O
     .
 Q020-SQL-ERROR-EXIT.
     EXIT.

 Q030-NOT-FOUND.
*************************************************************************
*  NOT FOUND PROCESSING                                               *
*************************************************************************

*    MOVE 'Q030' TO WORK-PARA

     SET  WS-EDIT-ERROR      TO  TRUE

     MOVE WS-B191-NUM              TO  MSGNUMO
                                       STC-HELP-MSG
     MOVE WS-B191-MSG              TO  MSG1O

     MOVE 'Q030 NOT FOUND '   TO  MSG1O
     MOVE BEN-SERIAL-NUM      TO  MSG1O (17 : 15)
     MOVE WORK-PARA           TO  MSG1O (34 : 8)
```

```
     .                                                                  0
 Q030-NOT-FOUND-EXIT.                                                   0
     EXIT.                                                              0
/                                                                   00136800
 Q300-GET-DISP-DATE.
*****************************************************************  00137000
*  INTERPETS ERROR AND DISPLAYS ON SCREEN                       *  00137100
*****************************************************************  00137300
                                                                    00137400
*    MOVE  'Q300    '                    TO  WORK-PARA             00063400
                                                                    00064600
     MOVE  WS-ODT-BYTE (1)          TO  WS-B2-LO                    00064700
     SUBTRACT  100  FROM  WS-BIN-2N                                 00064800
     MOVE  WS-BIN-2N                TO  WS-DT-CC                    00064900
     MOVE  WS-ODT-BYTE (2)          TO  WS-B2-LO                    00065000
     SUBTRACT  100  FROM  WS-BIN-2N                                 00065100
     MOVE  WS-BIN-2N                TO  WS-DT-YY                    00065200
     MOVE  WS-ODT-BYTE (3)          TO  WS-B2-LO                    00065300
     MOVE  WS-BIN-2N                TO  WS-DT-MM                    00065400
     MOVE  WS-ODT-BYTE (4)          TO  WS-B2-LO                    00065500
     MOVE  WS-BIN-2N                TO  WS-DT-DD                    00065600
     MOVE  WS-ODT-BYTE (5)          TO  WS-B2-LO                    00065700
     SUBTRACT  1    FROM  WS-BIN-2N                                 00064800
     MOVE  WS-BIN-2N                TO  WS-DT-HH                    00065800
     MOVE  WS-ODT-BYTE (6)          TO  WS-B2-LO                    00065900
     SUBTRACT  1    FROM  WS-BIN-2N                                 00064800
     MOVE  WS-BIN-2N                TO  WS-DT-MI                    00066000
     MOVE  WS-ODT-BYTE (7)          TO  WS-B2-LO                    00066100
     SUBTRACT  1    FROM  WS-BIN-2N                                 00064800
     MOVE  WS-BIN-2N                TO  WS-DT-SS                    00066200
                                                                    00137400
* MOVE FOR SCREEN DISPLAYS
     MOVE  WS-DT-CC                 TO  WS-DD-CC                    00066200
     MOVE  WS-DT-YY                 TO  WS-DD-YY                    00066200
     MOVE  WS-DT-MM                 TO  WS-DD-MM                    00066200
     MOVE  WS-DT-DD                 TO  WS-DD-DD                    00066200
     MOVE  WS-DT-HH                 TO  WS-TD-HH                    00066200
     MOVE  WS-DT-MI                 TO  WS-TD-MI                    00066200
     MOVE  WS-DT-SS                 TO  WS-TD-SS                    00066200
     .
 Q300-GET-DISP-DATE-EXIT.
     EXIT.

 Q350-DISP-TO-ORA.                                                  00079100
*************************************************************
*****  CONVERT DISPLAY DATE/TIME TO ORACLE DATE         ******
*************************************************************
*    MOVE 'Q350'                  TO    WORK-PARA.
                                                                    00137400
* MOVE FOR SCREEN DISPLAYS
     MOVE  WS-DD-CC                 TO  WS-DT-CC                    00066200
     MOVE  WS-DD-YY                 TO  WS-DT-YY                    00066200
     MOVE  WS-DD-MM                 TO  WS-DT-MM                    00066200
     MOVE  WS-DD-DD                 TO  WS-DT-DD                    00066200
     MOVE  WS-TD-HH                 TO  WS-DT-HH                    00066200
     MOVE  WS-TD-MI                 TO  WS-DT-MI                    00066200
     MOVE  WS-TD-SS                 TO  WS-DT-SS                    00066200
```

```
                                                                        00064600
           PERFORM  Q380-X14-TO-ORA THRU                                00079100
                    Q380-X14-TO-ORA-EXIT                                00079100
           .
       Q350-DISP-TO-ORA-EXIT.                                           00079100
           EXIT.

       Q380-X14-TO-ORA.                                                 00079100
      ****************************************************************
      *****  CONVERT X14 DATE FORMAT   TO ORACLE DATE          ******
      ****************************************************************
      *    MOVE 'Q380'                    TO    WORK-PARA.
                                                                        00064600
      *  CONVERT THE FIELDS
           MOVE  ZERO                        TO  WS-BIN-2N             00064700
           MOVE  WS-DT-CC                    TO  WS-BIN-2N             00064700
           ADD       100  TO    WS-BIN-2N                             00064800
           MOVE  WS-B2-LO                    TO  WS-ODT-BYTE (1)       00064700
           MOVE  ZERO                        TO  WS-BIN-2N             00064700
           MOVE  WS-DT-YY                    TO  WS-BIN-2N             00064700
           ADD       100  TO    WS-BIN-2N                             00064800
           MOVE  WS-B2-LO                    TO  WS-ODT-BYTE (2)       00064700
           MOVE     ZERO                     TO  WS-BIN-2N             00064700
           MOVE  WS-DT-MM                    TO  WS-BIN-2N             00065300
           MOVE  WS-B2-LO                    TO  WS-ODT-BYTE (3)       00065300
           MOVE  WS-DT-DD                    TO  WS-BIN-2N             00065300
           MOVE  WS-B2-LO                    TO  WS-ODT-BYTE (4)       00065300
           MOVE  WS-DT-HH                    TO  WS-BIN-2N             00065300
           ADD       1    TO    WS-BIN-2N                             00064800
           MOVE  WS-B2-LO                    TO  WS-ODT-BYTE (5)       00065300
           MOVE  WS-DT-MI                    TO  WS-BIN-2N             00065300
           ADD       1    TO    WS-BIN-2N                             00064800
           MOVE  WS-B2-LO                    TO  WS-ODT-BYTE (6)       00065300
           MOVE  WS-DT-SS                    TO  WS-BIN-2N             00065300
           ADD       1    TO    WS-BIN-2N                             00064800
           MOVE  WS-B2-LO                    TO  WS-ODT-BYTE (7)       00065300
           .
       Q380-X14-TO-ORA-EXIT.                                            00079100
           EXIT.

       R050-SELECT-CURRENT-ROW.                                         00013400
      ****************************************************************
      *****  SELECTS ROW USING THE CURRENT SCREENS KEY          ******
      ****************************************************************

      *    MOVE 'R050'                    TO    WORK-PARA.
                                                                        00013400
      *  SELECT CURRENT ROW FROM BENEFITS
           EXEC SQL SELECT
-INC CWELB800                                                           %
            INTO
-INC CWELB805                                                           %
            FROM  MCHECK.BENEFITS
             WHERE  SERIAL_NUM       =  RPAD(:BEN-SERIAL-NUM,15,' ')
               AND  CUSTOMER_ID      =  RPAD(:BEN-CUSTOMER-ID,14,' ')
               AND  BENEFIT_TYPE     =  RPAD(:BEN-BENEFIT-TYPE,5,' ')  00013400
               AND  EFFECTIVE_DATE   =  :BEN-EFFECTIVE-DATE
```

```
           END-EXEC

           PERFORM Q000-SQLCODE-PROCESS THRU                              00136900
                   Q000-SQLCODE-PROCESS-EXIT                              00136900

           IF  NOT ORA-SQL-SUCCESSFUL                                     00073800
             SET  WS-EDIT-ERROR  TO  TRUE                                 00136900
           END-IF                                                         00073800
           .
        R050-SELECT-CURRENT-ROW-EXIT.                                     00013400
           EXIT.

        R100-SEL-UPD-CURR-ROW.                                            00013400
       ***************************************************************
       *****  SELECTS AND UPDATES THE CURRENT ROW ON BENEFITS  ******
       ***************************************************************

       *    MOVE 'R100'                  TO    WORK-PARA.

       *  SELECT CURRENT ROW WITH UPDATE FROM BENEFITS
           EXEC SQL SELECT
-INC CWELB800                                                              %
           INTO
-INC CWELB805                                                              %
           FROM  MCHECK.BENEFITS
            WHERE  SERIAL_NUM       =  RPAD(:BEN-SERIAL-NUM,15,' ')
              AND  CUSTOMER_ID      =  RPAD(:BEN-CUSTOMER-ID,14,' ')
              AND  BENEFIT_TYPE     =  RPAD(:BEN-BENEFIT-TYPE,5,' ')   00013400
              AND  EFFECTIVE_DATE   =  :BEN-EFFECTIVE-DATE
           FOR UPDATE OF                                                  00017200
-INC CWELB800                                                              %
           END-EXEC

           PERFORM Q000-SQLCODE-PROCESS THRU                              00136900
                   Q000-SQLCODE-PROCESS-EXIT                              00136900

           IF  NOT ORA-SQL-SUCCESSFUL                                     00073800
             SET  WS-EDIT-ERROR  TO  TRUE                                 00136900
           END-IF                                                         00073800
           .
        R100-SEL-UPD-CURR-ROW-EXIT.                                       00013400
           EXIT.

        R140-UPD-HOLD-COLS.
       ***************************************************************
       *****   MOVE HOLD CONTROL DATA TO HOLD COLUMNS          ******
       ***************************************************************
       *    MOVE 'R140'                  TO    WORK-PARA.

           MOVE STU-USER-ID             TO    BEN-HOLD-USER-ID
                                                                          00062900
           EXEC SQL SELECT
                 SYSDATE INTO :WS-PROCESS-DATE
              FROM  DUAL
           END-EXEC
           MOVE  WS-PROCESS-DATE        TO   BEN-HOLD-DT-TM
           .
```

```
 R140-UPD-HOLD-COLS-EXIT.
     EXIT.

 R145-UPD-HOLD-SCRN-DATA.
*****************************************************************
*****   MOVE HOLD CONTROL DATA TO HOLD COLUMNS           ******
*****************************************************************
*    MOVE 'R145'                    TO    WORK-PARA.

     MOVE  BENEFITS-ROW              TO  STU-BEN-ROW                 00088800
     MOVE  -1                        TO  CARD-SER1L                  00065200

     MOVE  BEN-HOLD-CDE              TO  HOLD-STO                    00013400
     EVALUATE  TRUE                                                  00013400
      WHEN  BEN-NO-EXTERNAL-HOLD                                     00013400
       MOVE  '  AVAILABLE   '      TO  HOLD-DESCO                  00013400
                                                                     00013400
      WHEN  BEN-HOLD-THIS-BENEFIT                                    00013400
       MOVE  'HOLD - SER #    '    TO  HOLD-DESCO                  00013400
                                                                     00013400
      WHEN  BEN-HOLD-ALL-CUST-BENEFITS                               00013400
       MOVE  'HOLD - CUSTOMER'     TO  HOLD-DESCO                  00013400
                                                                     00013400
      WHEN  OTHER                                                    00013400
       MOVE  '  NOT DEFINED  '     TO  HOLD-DESCO                  00013400
     END-EVALUATE                                                    00013400
                                                                     00013400
     MOVE  BEN-HOLD-DT-TM            TO  WS-ORA-DT-TM
     PERFORM  Q300-GET-DISP-DATE THRU
              Q300-GET-DISP-DATE-EXIT
     MOVE  WS-DATE-DISP-10           TO  HOLD-DATEO
     MOVE  WS-TIME-DISP-8            TO  HOLD-TIMEO
                                                                     00137400
     MOVE  BEN-HOLD-USER-ID          TO  HOLD-UIDO
     .
 R145-UPD-HOLD-SCRN-DATA-EXIT.
     EXIT.

 R150-UPDATE-CURR-ROW.
*****************************************************************
*****   UPDATES THE SELECTED BENEFITS ROW               ******
*****************************************************************
*    MOVE 'R150'                    TO    WORK-PARA.

     EXEC  SQL  UPDATE  MCHECK.BENEFITS                              00036600
       SET                                                           00036600
           HOLD_DT_TM        =  :BEN-HOLD-DT-TM,
           HOLD_CDE          =  :BEN-HOLD-CDE,
           HOLD_USER_ID      =  :BEN-HOLD-USER-ID
       WHERE                                                         00036600
             SERIAL_NUM        =   RPAD(:BEN-SERIAL-NUM,15,' ')      00036600
         AND CUSTOMER_ID       =   RPAD(:BEN-CUSTOMER-ID,14,' ')     00036600
         AND BENEFIT_TYPE      =   RPAD(:BEN-BENEFIT-TYPE,5,' ')     00013400
         AND EFFECTIVE_DATE    =   :BEN-EFFECTIVE-DATE               00036600
     END-EXEC                                                        00036600

     PERFORM Q000-SQLCODE-PROCESS THRU                               00136900
```

```
           Q000-SQLCODE-PROCESS-EXIT                                    00136900

     IF  NOT ORA-SQL-SUCCESSFUL                                         00073800
       SET  WS-EDIT-ERROR  TO  TRUE                                     00136900
     END-IF                                                             00073800
     .
 R150-UPDATE-CURR-ROW-EXIT.
     EXIT.

 R200-STARTBR-INDICES.
*********************************************************************
*****  ROUTINE TO START BROWSE BENEFIT INDICES                  ****
*********************************************************************
     MOVE 'R200'                    TO    WORK-PARA

     EVALUATE  TRUE
      WHEN  STU-SEARCH-SER-NUM
       SET  WS-MFB1-NDX-SERIAL-NUM  TO  TRUE
       MOVE BEN-SERIAL-NUM          TO    WS-MFB1-X1-SERIAL-NUM
       MOVE BEN-CUSTOMER-ID         TO    WS-MFB1-X1-CUSTOMER-ID
       MOVE BEN-BENEFIT-TYPE        TO    WS-MFB1-X1-BENEFIT-TYPE
       MOVE  BEN-EFFECTIVE-DATE     TO  WS-ORA-DT-TM
       PERFORM  Q300-GET-DISP-DATE THRU
                Q300-GET-DISP-DATE-EXIT
       MOVE  WS-DATE-TIME           TO    WS-MFB1-X1-EFFECTIVE-DATE
       MOVE BEN-MFG-SERIAL-NUM      TO    WS-MFB1-X1-MFG-SERIAL-NUM
       MOVE WS-MFB1-KEY             TO    WS-SAVE-MFB1-KEY

      WHEN  STU-SEARCH-MFG-SER-NUM
       SET  WS-MFB1-NDX-MFG-SERIAL-NUM  TO  TRUE
       MOVE BEN-SERIAL-NUM          TO    WS-MFB1-X2-SERIAL-NUM
       MOVE BEN-CUSTOMER-ID         TO    WS-MFB1-X2-CUSTOMER-ID
       MOVE BEN-BENEFIT-TYPE        TO    WS-MFB1-X2-BENEFIT-TYPE
       MOVE  BEN-EFFECTIVE-DATE     TO  WS-ORA-DT-TM
       PERFORM  Q300-GET-DISP-DATE THRU
                Q300-GET-DISP-DATE-EXIT
       MOVE  WS-DATE-TIME           TO    WS-MFB1-X2-EFFECTIVE-DATE
       MOVE BEN-MFG-SERIAL-NUM      TO    WS-MFB1-X2-MFG-SERIAL-NUM
       MOVE WS-MFB1-KEY             TO    WS-SAVE-MFB1-KEY

      WHEN  STU-SEARCH-CUST-ID
       SET  WS-MFB1-NDX-CUSTOMER-ID  TO  TRUE
       MOVE BEN-SERIAL-NUM          TO    WS-MFB1-X3-SERIAL-NUM
       MOVE BEN-CUSTOMER-ID         TO    WS-MFB1-X3-CUSTOMER-ID
       MOVE BEN-BENEFIT-TYPE        TO    WS-MFB1-X3-BENEFIT-TYPE
       MOVE  BEN-EFFECTIVE-DATE     TO  WS-ORA-DT-TM
       PERFORM  Q300-GET-DISP-DATE THRU
                Q300-GET-DISP-DATE-EXIT
       MOVE  WS-DATE-TIME           TO    WS-MFB1-X3-EFFECTIVE-DATE
       MOVE BEN-MFG-SERIAL-NUM      TO    WS-MFB1-X3-MFG-SERIAL-NUM
       MOVE WS-MFB1-KEY             TO    WS-SAVE-MFB1-KEY
     END-EVALUATE

     EXEC CICS
          STARTBR
          FILE   ('DEFMFB1')
          RIDFLD (WS-MFB1-KEY)
```

```
          GTEQ
          RESP    (WS-RESP)
      END-EXEC.

      SET  WS-STARTBR  TO  TRUE                                          0
      PERFORM  R280-CHECK-RESP THRU
               R280-CHECK-RESP-EXIT
      .
 R200-STARTBR-INDICES-EXIT.
      EXIT.

 R280-CHECK-RESP.
*****************************************************************
*****  CHECKS WS-RESP FOR GOOD FUNCTION                    *******
*****************************************************************
*     MOVE 'B280'                    TO    WORK-PARA

      EVALUATE  TRUE
       WHEN  WS-RESP  =  DFHRESP(NORMAL)
        EVALUATE  TRUE
         WHEN  STU-SEARCH-SER-NUM
           IF  NOT  WS-MFB1-NDX-SERIAL-NUM
             PERFORM R285-END-OF-SEARCH THRU
                     R285-END-OF-SEARCH-EXIT
           END-IF

         WHEN  STU-SEARCH-MFG-SER-NUM
           IF  NOT  WS-MFB1-NDX-MFG-SERIAL-NUM
             PERFORM R285-END-OF-SEARCH THRU
                     R285-END-OF-SEARCH-EXIT
           END-IF

         WHEN  STU-SEARCH-CUST-ID
           IF  NOT  WS-MFB1-NDX-CUSTOMER-ID
             PERFORM R285-END-OF-SEARCH THRU
                     R285-END-OF-SEARCH-EXIT
           END-IF

         WHEN  OTHER
           SET  WS-EDIT-ERROR      TO  TRUE
           MOVE WS-B190-NUM              TO  MSGNUMO
                                             STC-HELP-MSG
           MOVE WS-B190-MSG              TO  MSG1O
           MOVE  STU-SEARCH-FLAG         TO  MSG1O (22 : 1)
        END-EVALUATE

       WHEN  WS-RESP  =  DFHRESP(NOTFND)
       WHEN  WS-RESP  =  DFHRESP(ENDFILE)
        PERFORM R285-END-OF-SEARCH THRU
                R285-END-OF-SEARCH-EXIT

       WHEN  OTHER
        SET  WS-EDIT-ERROR      TO  TRUE
        MOVE WS-B189-NUM              TO  MSGNUMO
                                          STC-HELP-MSG
        MOVE WS-B189-MSG              TO  MSG1O
        MOVE  WS-RESP                 TO  WS-SQLCODE-DISP
```

```
       MOVE  WS-SQLCODE-DISP          TO  MSG1O (18 : 5)
       MOVE  WORK-PARA                TO  MSG1O (30 : 8)
       MOVE  WS-FUNCTION-ID           TO  MSG1O (39 : 9)                  0
       MOVE   'KEY: '                 TO  MSG3O                           0
       MOVE  WS-MFB1-KEY              TO  MSG3O (6 : 40)                  0
     END-EVALUATE
     .
 R280-CHECK-RESP-EXIT.
     EXIT.

 R285-END-OF-SEARCH.
*****************************************************************
*****  SETS END OF SEARCH CONDITION INCLUDING MESSAGE      *******
*****************************************************************
*    MOVE 'R285'                   TO    WORK-PARA

     SET  WS-END-SEARCH  TO  TRUE

     MOVE WS-B180-NUM              TO  MSGNUMO                     00063200
                                       STC-HELP-MSG                00063300
     MOVE WS-B180-MSG              TO  MSG1O

     EVALUATE  TRUE                                                00083400
      WHEN  WS-GET-NEXT-ROW                                        00083400
      WHEN  STU-INITIAL-SCREEN                                     00083400
      WHEN  WS-GET-CURRENT-ROW                                     00083400
       MOVE  'LAST '               TO  MSG1O (1 : 5)               00083400
                                                                   00083400
      WHEN  WS-GET-PREVIOUS-ROW                                    00083400
       MOVE  'FIRST'               TO  MSG1O (1 : 5)               00083400
                                                                   00083400
      WHEN   OTHER                                                 00083400
       MOVE  'OTHER'               TO  MSG1O (1 : 5)               00083400
     END-EVALUATE                                                  00083400
     .
 R285-END-OF-SEARCH-EXIT.
     EXIT.

 R300-READNEXT-INDICES.
*****************************************************************
*****  READS NEXT BENEFIT INDEX                          *******
*****************************************************************

     EXEC CICS
          READNEXT
          FILE   ('DEFMFB1')
          INTO   (WS-MFB1-RECORD)
          RIDFLD (WS-MFB1-KEY)
          RESP   (WS-RESP)
     END-EXEC

     SET  WS-READNEXT  TO  TRUE                                    0
     PERFORM  R280-CHECK-RESP THRU
              R280-CHECK-RESP-EXIT
     .
 R300-READNEXT-INDICES-EXIT.
     EXIT.
```

```
 R350-READPREV-INDICES.
*****************************************************************
*****  READS PREVIOUS BENEFIT INDEX                          *******
*****************************************************************
*    MOVE 'B350'                  TO    WORK-PARA

     EXEC CICS
          READPREV
          FILE   ('DEFMFB1')
          INTO   (WS-MFB1-RECORD)
          RIDFLD (WS-MFB1-KEY)
          RESP   (WS-RESP)
     END-EXEC.

     SET  WS-READPREV  TO  TRUE                                          0
     PERFORM  R280-CHECK-RESP THRU
              R280-CHECK-RESP-EXIT
     .
 R350-READPREV-INDICES-EXIT.
     EXIT.


 S300-SYSLOG.
*****************************************************************
*****                                                      ******
*****  THIS ROUTINE WILL BUILD A MESSAGE TO BE PLACED      ******
*****  ON THE SYSTEM LOG, AND IT WILL LINK TO THE          ******
*****  AFC SYSTEM LOGGER PROGRAM.                          ******
*****                                                      ******
*****************************************************************
*    MOVE 'S300'                  TO    WORK-PARA.

     MOVE 0                       TO    DTC-PROCESS-DIR.


     PERFORM X400-GET-DATE-TIME
        THRU X400-GET-DATE-TIME-EXIT.

     MOVE DTC-GMT                 TO    SLF-TRANS-DT-TM.
     MOVE EIBTRMID                TO    SLF-TERMINAL-ID.
     MOVE WS-PROGRAM-ID           TO    SLF-PROGRAM-ID.
     MOVE STC-USER-ID             TO    SLF-INITIATOR-ID.
     MOVE EIBTASKN                TO    SLF-INITIATOR-TASK-ID
                                        SLF-PROCESSOR-TASK.

     EXEC CICS LINK
               PROGRAM  ('CWAC3100')
               COMMAREA (SYSTEM-LOGGER-COMMAREA)
               LENGTH   (LENGTH OF SYSTEM-LOGGER-COMMAREA)
               RESP     (WS-RESP)
     END-EXEC.


 S300-SYSLOG-EXIT.
     EXIT.
/
```

```
       W200-WRITE-BEFORE-IMAGE.

           SET  SLC-MESSAGE-IN-COMMAREA
                                       TO    TRUE.
           SET  SLF-UPDATE-RECORD-BEFORE
                                       TO    TRUE.
********** MOVE LENGTH OF EF3-CARDHOLDER
**********                             TO    SLC-MESSAGE-LENGTH.
********** MOVE EF3-CARDHOLDER         TO    SLF-MESSAGE-VALUE.

           PERFORM S300-SYSLOG
              THRU S300-SYSLOG-EXIT.

           IF WS-RESP NOT = DFHRESP(NORMAL)
              MOVE WS-B183-NUM                 TO MSGNUMO          00103000
              MOVE WS-B183-MSG                 TO MSG1O            00103000
              MOVE WS-PFKEY-MSG1               TO PFKEY-MSGO       00089200

           END-IF.

       W200-WRITE-BEFORE-IMAGE-EXIT.
           EXIT.

      /                                                              00272600
       X100-WRITE-QUEUE.                                             00272700
      ***************************************************************00272800
      *  THIS PARAGRAPH WRITES DATA TO THE TEMPORARY STORAGE QUEUE.   *00272900
      ***************************************************************00273000
                                                                     00273100
           MOVE 'X100' TO WORK-PARA.                                 00273200
                                                                     00273400
           COMPUTE SQA-QUE-NUM = STC-MAX-Q-ON-ENTRY (STC-PGM-TR-CNT) + 100273500
                                                                     00273600
           MOVE EIBTRMID                TO SQA-TERMINAL-ID.          00273700
           MOVE LENGTH OF WS-QUEUE-DATA TO WS-RECORD-LENGTH.         00273800
                                                                     00273900
           EXEC CICS WRITEQ TS                                       00274000
                     QUEUE  (SQA-QUE)                                00274100
                     FROM   (WS-QUEUE-DATA)                          00274200
                     LENGTH (WS-RECORD-LENGTH)                       00274300
                     ITEM   (WS-ITEM)                                00274400
                     REWRITE                                         00274500
                     RESP   (WS-RESP)                                00274600
           END-EXEC.                                                 00274700
                                                                     00274800
           IF  WS-RESP = DFHRESP (QIDERR)                            00274900
               MOVE +1                  TO WS-ITEM                   00275000
               EXEC CICS WRITEQ TS                                   00275100
                         QUEUE  (SQA-QUE)                            00275200
                         FROM   (WS-QUEUE-DATA)                      00275300
                         LENGTH (WS-RECORD-LENGTH)                   00275400
                         ITEM   (WS-ITEM)                            00275500
                         RESP   (WS-RESP)                            00275600
               END-EXEC                                              00275700
           END-IF.                                                   00275800
                                                                     00275900
           IF  WS-RESP NOT = DFHRESP (NORMAL)                        00276000
```

```
         GO TO SKABEXIT-LEVEL2                                          00276100
     END-IF.                                                            00276200
                                                                        00276300
 X100-WRITE-QUEUE-EXIT.                                                 00276400
     EXIT.                                                              00276500
                                                                        00276600
 X200-READ-QUEUE.                                                       00276700
*****************************************************************     00276800
*  THIS PARAGRAPH READS DATA FROM THE TEMPORARY STORAGE QUEUE.  *       00276900
*****************************************************************     00277000
                                                                        00277100
     MOVE 'X200' TO WORK-PARA.                                          00277200
                                                                        00277400
     COMPUTE SQA-QUE-NUM = STC-MAX-Q-ON-ENTRY (STC-PGM-TR-CNT) + 1      00277500
                                                                        00277600
     MOVE EIBTRMID                 TO SQA-TERMINAL-ID.                  00277700
     MOVE LENGTH OF WS-QUEUE-DATA TO WS-RECORD-LENGTH.                  00277800
                                                                        00277900
     EXEC CICS READQ TS                                                 00278000
               QUEUE  (SQA-QUE)                                         00278100
               INTO   (WS-QUEUE-DATA)                                   00278200
               LENGTH (WS-RECORD-LENGTH)                                00278300
               ITEM   (WS-ITEM)                                         00278400
               RESP   (WS-RESP)                                         00278500
     END-EXEC.                                                          00278600
                                                                        00278700
     IF  WS-RESP NOT = DFHRESP (NORMAL)                                 00278800
         GO TO SKABEXIT-LEVEL2                                          00278900
     END-IF.                                                            00279000
                                                                        00279100
 X200-READ-QUEUE-EXIT.                                                  00279200
     EXIT.                                                              00279300
/                                                                       00279400
 X300-READ-REFER.                                                       00279500
*****************************************************************     00279600
*  THIS PARAGRAPH READS DATA FROM THE REFERENCE TABLE.          *       00279700
*****************************************************************     00279800
                                                                        00279900
     MOVE 'X300' TO WORK-PARA.                                          00280000
                                                                        00280200
     MOVE REFERENCE-FILE-VALUES           TO RF-TABLE-ID.               00280300
*    MOVE WS-BIN-2C                       TO WS-TINY-INT.               00280400
*    MOVE WS-TINY-CHR                     TO RF-TABLE-ENTRY-ID.         00280500
     MOVE LENGTH OF RF-REFERENCE-RCD      TO WS-RECORD-LENGTH.          00280600
                                                                        00280700
     EXEC CICS READ                                                     00280800
               FILE      ('DEFMF03')                                    00280900
               INTO      (RF-REFERENCE-RCD)                             00281000
               LENGTH    (WS-RECORD-LENGTH)                             00281100
               RIDFLD    (RF-KEY)                                       00281200
               GENERIC                                                  00281300
               EQUAL                                                    00281400
               KEYLENGTH (5)                                            00281500
               RESP      (WS-RESP)                                      00281600
     END-EXEC.                                                          00281700
                                                                        00281800
     IF  WS-RESP = DFHRESP (NORMAL)                                     00281900
```

```
           AND RF-ENTRY-ACTIVE                                          00282000
               MOVE RF-ENTRY-DESCRIPTION (1:15) TO WS-DESC              00282100
           ELSE                                                         00282200
               MOVE 'DESC NOT FOUND'            TO WS-DESC              00282300
           END-IF.                                                      00282400
                                                                        00282500
         X300-READ-REFER-EXIT.                                          00282600
           EXIT.                                                        00282700
       /
         X400-GET-DATE-TIME.
         ************************************************************
         *****                                                  ****
         ***** ROUTINE TO LINK TO GMT CONVERSION PROGRAM TO GET  ****
         ***** DATE AND TIME                                    ****
         *****                                                  ****
         ************************************************************
           MOVE 'X300'                  TO    WORK-PARA.

           MOVE 0                       TO    DTC-RTN-CODE.

           EXEC CICS LINK
                     PROGRAM   ('CWAX3000')
                     COMMAREA  (DATE-TIME-CONVERSION)
                     LENGTH    (LENGTH OF DATE-TIME-CONVERSION)
                     RESP      (WS-RESP)
           END-EXEC.

           IF WS-RESP NOT = DFHRESP(NORMAL)
              MOVE WS-B186-NUM              TO MSGNUMO
                                               STC-HELP-MSG             00063300
              MOVE WS-B186-MSG              TO MSG1O
              MOVE WS-RESP                  TO WS-RESP-9-DISPLAY
              MOVE WS-RESP-9-DISPLAY        TO MSG1O (28 : 2)
              MOVE WS-PFKEY-MSG1            TO PFKEY-MSGO              00089200

              SET  STC-EDIT-ERROR           TO    TRUE

           ELSE
              IF DTC-RTN-CODE > 5
                 MOVE WS-B185-NUM              TO MSGNUMO
                                                  STC-HELP-MSG          00063300
                 MOVE WS-B185-MSG              TO MSG1O
                 MOVE WS-PFKEY-MSG1            TO PFKEY-MSGO            00089200

                 SET  STC-EDIT-ERROR           TO    TRUE
              END-IF
           END-IF
           .
         X400-GET-DATE-TIME-EXIT.
           EXIT.
       /                                                                00284200
-INC CWPL9020                                                           0028430%
       /                                                                00284400
-INC CWPL9030                                                           0028450%
       /                                                                00284600
-INC CWXL9910                                                           0028470%
       /                                                                00284800
```

-INC CWXL9990

0028490%

APPENDIX I

```
/*================================================================

                Copyright 1988   Cubic Western Data
                     San Diego, California

   ================================================================


   File:             SMADDEB.H

   Description:      Common literal definitions  and prototypes



   ================================================================*/

#ifndef SMADDEB_HDR
#define SMADDEB_HDR

#ifndef SHMSG005
#include "shmsg005.h"
#endif


#ifndef SHDSMPUB_H
#include "shdsmpub.h"
#endif

#ifndef SMADFEP_HDR
#include "smadfep.h"
#endif

                /* Name of queue.*/
#define DEB_QUEUE_NAME  "\\queues\\Deb.que"


typedef struct
{
   UCHAR    uchRefNo[RETRIEVAL_REF_NUMBER_MAX];
} DC_SECURE_DEL_REQUEST;


/* Message from SMADFEP thread to response from device for SECURE or DELETE DC
TRANS msgs */
typedef struct
{
    UCHAR   uchMachine;
    UCHAR   uchRefNo[RETRIEVAL_REF_NUMBER_MAX];
    UCHAR   uchResultCode;
} DC_SECURE_DEL_RESPONSE;



/* EUCx msg block from device via FEP thread to DBCD Thread */
typedef struct
```

```
{
    UCHAR   uchMachine;
    TX_MULTI_BLOCK_MSG  EUC_Msg;
} DC_TRANS_RESPONSE;



/* Mail msg block from SHDEBCRED module via SendToDevice function */
/* When the SHDEBCRD module receivs a MACK fromthe CC after a transaction
   has been sent and received, the Mack is passed to the
   SendToHost function which send sthe DC_DEL_TRANS to the DCThread.
   The DCThread  will send the message via the FEP Thread to delete the
   specified Debit/Credit transaction. */
typedef struct
{
    UCHAR   uchMachine;
    UCHAR   uchMackType[MSG_ID_TYPE_MAX]; /* 4 byte msg type (EUC3/4/5 */
    ULONG   ulCICS;   /* CICS Number used in header of CC message */
    UCHAR   auchRefNum[RETRIEVAL_REF_NUMBER_MAX]; /* reference number */
} DC_DEL_TRANS;


/* External EUC2 or MACK message received from SHDEBCRE thread */
typedef struct
{
    MSG_C_HEADER    tHDR;
    UCHAR           uchData[MSG_MAX_LENGTH - sizeof(MSG_C_HEADER)];
} DC_EXTERNAL_EUC2_MSG;


/* Mail que message used to send message from SendToDevice to DC_THREAD */
typedef struct
{
    UCHAR           uchMachine;
    USHORT          usSize;
    MSG_C_HEADER    tHDR;
    UCHAR           uchData[MSG_MAX_LENGTH - sizeof(MSG_C_HEADER)];
} DC_EXTERNAL_EUC_MSG;


#endif
```

APPENDIX J

```
/*====================================================================

                Copyright 1995   Cubic Western Data
                     San Diego, California

   ====================================================================


   File:             SMADDEB.C

   Description:



   Date:             22 June 1997

   By:               DPY

   ====================================================================*/


         /*  I N C L U D E   F I L E S  */

#define INCL_BASE
#include <stdio.h>
#include <stdlib.h>
#include <io.h>
#include <os2.h>
#include <string.h>
#include <stddef.h>
#include <process.h>

#include "smadcmn.h"
#include "smaderr.h"
#include "smadmsg.h"
#include "smadmsq.h"
#include "smadadm.h"
#include "smadfep.h"
#include "shlib000.h"
#include "shmsg004.h"
#include "shmsg005.h"
#include "shmsg006.h"
#include "shapc000.h"
#include "smadprot.h"
#include "shdsmpub.h"
#include "shdebcre.h"
#include "smadccx.h"
#include "smaddeb.h"



         /*  L I T E R A L     D E C L A R A T I O N S  */
#define DC_EUC2_MAX_SIZE        512
#define DC_TRANS_MAX_SIZE       512  /* for temp storage of EUC3, EUC4, or EUC5
*/

/* Return codes and defines used only in this module */
```

```
#define DC_EUC2_MAX_WAIT        12000    /* 12 seconds */
#define DC_MAX_WAIT             10000   /* 10 seconds */
#define DC_EBCDIC_ZERO          240     /* HEX F0 value for EBCDIC character
ZERO */
#define DC_EBCDIC_B             0xC2    /* HEX C2 value for EBCDIC character 'B'
*/


#define DC_EUC2_TIMEOUT         99  /* local msg codes for this module */
#define DC_SECURE_TIMEOUT       98
#define DC_DELETE_TIMEOUT       97
#define DC_EUC2_NAK             96
#define DC_SECURE_NAK           95
#define DC_DELETE_NAK           94
#define DC_RESEND_TIMEOUT       93
#define DC_TRANS_TIMEOUT        92


#define SIX_HOURS             21600   /* 60*60*6 */

#define DC_STACKSIZE            100000

#define DC_DEL_MAX_RETRY 1
#define DC_EUC2_MAX_RETRY 1
#define DC_MAX_RETRY     2  /* max times to resend after timeouts waiting for
response */
#define DC_MAX_RESEND_RETRY  10 /* try more for resend commands */

/* offsets into EUC external packed byte messages */
#define DC_EUC1_RET_REF_OFFSET 174  /* offset from start of EUC1/EUC6 to ret.ref
field */
#define DC_EUC3_RET_REF_OFFSET 178  /* offset from start of EUC3 to ret.ref
field */
#define DC_EUC5_RET_REF_OFFSET 351  /* offset from start of EUC5      to ret.ref
field */
#define DC_EUB1_RET_REF_OFFSET  39
#define DC_EUB5_RET_REF_OFFSET  39
#define DC_EUB3_RET_REF_OFFSET  39


/* States for device */
#define DC_IDLE            1
#define DC_TRANS_ACK       2
#define DC_WAITING_ACK     3

/* some constants for setting up transaction index */
#define DC_SERIAL_NUMBER_DIGITS 5
#define DC_SERIAL_NUMBER_SHIFT  10000000  /* low order 7 digits is GMT, high
digits for serial num */
#define DC_MSG_TYPE_SHIFT       10 /* decimal shift for EUC msg type (1 digit)
*/
#define DC_GMT_DIGITS_IN_LONG   10  /* number of decimal digits in a LONG viewed
in ASCII format */
#define DC_GMT_DIGITS           7

/* Used in   DBCRDT_Error */
#define DC_ERROR_MESSAGE    "DC Error:                                    "
```

```
#define DC_ERR_LINENO       10
#define DC_ERR_MSG_SZ       sizeof(DC_ERROR_MESSAGE)

/* Typedefs */
typedef enum DC_EVENTS
{   DC_EUC2_EVENT=0,
    DC_SECURE_EVENT,
    DC_DELETE_EVENT,
    DC_RESEND_EVENT,
    DC_MAX_EVENT
} DC_TIMEOUT_EVENTS;


typedef struct
{
    UCHAR                   uchBlock;
    UCHAR                   uchC2Block;
    USHORT                  usBytePtr;
    USHORT                  usState[DC_MAX_EVENT];
    USHORT                  usRetryCount[DC_MAX_EVENT];
    USHORT                  usSeqNum;
    USHORT                  usC2SeqNum;
    USHORT                  usTransIndex;
    USHORT                  usTryAgain;
    INT                     iMsgSize;
    INT                     iOrgSize; /* Save original size in case we have to
resend msg */
    UCHAR                   uchMackType[MSG_ID_TYPE_MAX];
    UCHAR                   EUC2[DC_EUC2_MAX_SIZE];  /*EUC2 in external format*/
    UCHAR                   uchExt_TRANS[DC_TRANS_MAX_SIZE];  /* EUC3,4 or 5
external format*/
    UCHAR                   uchSecureRefNo[RETRIEVAL_REF_NUMBER_MAX];
    UCHAR                   uchDeleteRefNo[RETRIEVAL_REF_NUMBER_MAX];
    ULONG                   ulErrorCount;
    ULONG                   ulCICS;
    ULONG                   ulEUC2_CICS;
    TIME_T                  ulTime[DC_MAX_EVENT];
} DC_DEVICE_STR;



        /* E X T E R N A L    D E C L A R A T I O N S */

extern PPVOID   SMADS_sel;


        /* G L O B A L   D E C L A R A T I O N S */
UCHAR   auchTmpMsg[80];  /* for debug string output */
HQUEUE  ulDEB_CRED_handle;
ULONG   ulDC_Debug      =   FALSE;
ULONG   ulEVDC_Debug    =   FALSE; /* Used for table handline */
static  DC_DEVICE_STR  tDevice_Str[MAX_NUM_VENDORS];
```

```
/* prototype */
VOID    ENV_CDECL    DCThread (PVOID dummy);
/*VOID    ENV_CDECL    FormatEUC_Msg (PVOID , USHORT );*/


/*****************************************************************************
* Module:   DBCRDT_Display
* Desc:     This displays messages sent to the CC for Debit/Credit Transactions
*
* Inputs:   Pointer to string to Display
* Outputs:  N/A
* Errors:   Default Error Processing.
*****************************************************************************/

VOID    DBCRDT_Display(PCHAR szDispStr)
{

    PCHAR pStr;

    /* strip of the \r\n characters from the string passed in */
    pStr = (PCHAR) memchr(szDispStr, '\r', 80);
    if (pStr != NULL)
    {
        *pStr = '\0';
    }


    SH_REPORT_EVENT(NOTIFY_EVENT,
                        MC_STATUS_NONE, MC_STATUS_IGNORE,
                        (APIRET) 0,
                       szDispStr);

}


/*****************************************************************************
* Module:   DBCRDR_DSM_Created
* Desc:     This function will not do anything for transit authority
*
* Inputs:   N/A
* Outputs:  N/A
* Errors:   Default Error Processing.
*****************************************************************************/

VOID    DBCRDT_DSM_Created(VOID)
{

    APIRET  error_ind;

    /* send message to DCThread*/
    error_ind = WriteToQ (&ulDEB_CRED_handle, SMADS_sel, DC,
                NULL, DC_MSG, DC_RESEND,
                0, 0);
    if (error_ind != 0)
    {
       SH_REPORT_EVENT (WARNING_EVENT, SH_STATUS_SWERR,
                             SH_STATUS_IGNORE, error_ind,
```

```
                                "Error Returned by WriteToQ");
    }
}


/*****************************************************************************
* Module:   DBCRDT_Error
* Desc:     This perform any error handling for Debit/Credit
*
* Inputs:   Error Information and the File and line that it occured
* Outputs:  N/A
* Errors:   Default Error Processing.
*****************************************************************************/

VOID    DBCRDT_Error(INT iErrorID, ULONG ulErrorCode,
                     PCHAR szFileName, SHORT sLineNo)

{
    static CHAR szMsg[120];


    sprintf (&szMsg[0], "%s%3i Code:%09x, Line:%4i, File:%s",
            "DC_ERROR:",iErrorID, ulErrorCode, sLineNo, szFileName);

    SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                    SH_STATUS_IGNORE, (ULONG) iErrorID,
                    szMsg);

}

/*****************************************************************************
* Module:   CredDebInit
* Desc:     This performs all initialization to the debit/credit thread
*
* Inputs:   N/A
* Outputs:  N/A
* Errors:   Default Error Processing.
*****************************************************************************/

APIRET DebCredInit (VOID)
{
    BOOL        rc;
    USHORT      Debit_Version;
    USHORT      Debit_Revision;
    ULONG       ulStatus;
    INT         iSecondsForResend = 180; /* 3 minutes check for messages to be
resent */
    INT         iHalfSecWaitsForMack; /* Used to init SHDEBCRE for how long to
wait
                                         for an EUC2 MACK from device (or this
SMADDEB
                                         thread in the case an EUC2 could not be
delivered
                                         to the device.*/
```

```
    /*  Note: must be slightly longer
              then 2*((DC_EUC2_MAX_RETRY+1)*DC_EUC2_MAX_WAIT/1000) */
    iHalfSecWaitsForMack = 8 + (2 * ( (DC_EUC2_MAX_RETRY+1) *
(DC_EUC2_MAX_WAIT/1000)));
    /* calculates to 56 (28 seconds) :
       2 retries after 12 second response waiting periods (16 seconds) plus 4
seconds extra*/

    /* create the queue for receiving messages from other threads */
    do
    {
        ulStatus = DosCreateQueue (&ulDEB_CRED_handle, 0, DEB_QUEUE_NAME);
        DosSleep (500L);
    }
    while (ulStatus != 0);


   /** Debit/Credit device Thread  **/
   if (_beginthread (DCThread, NULL, DC_STACKSIZE, NULL) == 1)
   {
     WriteString("DC thread creation error",20,3);
   }


   /** Initialize Debit/Credit Threads ***/
   Debit_Version = 1;
   Debit_Revision = 0;

   rc = DBCRDT_Init(Debit_Version, Debit_Revision,
                    iHalfSecWaitsForMack,
                    iSecondsForResend);


   return ((APIRET) rc);
}



/*****************************************************************************
* Module:    DC_Create_DSM_Index
* Desc:      Create lookup index and info field used by the DSM in the SHDBCRD
thread
*           for unique key for the transaction in the data storage manager
(DSM).
*           The DSM uses the MSG_C_HEADER CICS (4 byte integer) field. To make
*           this unique, we will use the vendor number and  transaction type (1-
6)
*           and the 7 digits of the
*           GMT from the EUCx message RETRIEVAL_REF_NUMBER field. (Note this
field
*           is 12 EBCDIC digits - 5 high order digits are the device serial
number,
*           the lower 7 digits are the GMT).
*           The 5 serial number digits in the RETRIEVAL_REF_NUMBER are put into
the 3
*           byte auchInfo field that is kept with the transaction and returned
by
```

```
*           the DSM in MACK messages.
*
*           The CICS key is:
*                               v,vcg,ggg,ggg
*               where   'vv' is the vendor number (0-15)
*                       'c'  is in the range 1 to 6 (the x from EUCx message
type)
*                       'ggggggg' is the 7 GMT digits from the
RETRIEVAL_REF_NUMBER
*
*
* Inputs:   pauchRefNum     Pointer to EBCDIC string for  RETRIEVAL_REF_NUMBER
*           uchMachine      machine number;
*           uchMsgType      message type (1 to 6 for EUC1 to EUC6)
* Outputs:  pauchInfo       Pointer to 3 byte string for machine serial number
(binary)
*           pulCICS         Pointer to long for unique key
*                           The CICS key is:
*                                v,vcg,ggg,ggg
*                           where   'vv' is the vendor number (0-15)
*                                   'c'  is in the range 1 to 6 (the x from EUCx
message type)
*                                   'ggggggg' is the 7 GMT digits from the
RETRIEVAL_REF_NUMBER
* Errors:   Default Error Processing.
*****************************************************************************/

VOID    DC_Create_DSM_Index (PUCHAR pauchRefNum, UCHAR uchMsgType,
                             UCHAR  uchMachine,  PUCHAR pauchInfo, PULONG
pulCICS)
{



    UCHAR   uchGMT[DC_GMT_DIGITS+1];
    UCHAR   uchSerialNum[DC_SERIAL_NUMBER_DIGITS+1];
    ULONG   ulTemp;


    /* extract GMT from retrieval reference number */
    LIB_cnv_ebcdic_to_ascii ((PUCHAR)pauchRefNum+DC_SERIAL_NUMBER_DIGITS,
                             (PUCHAR)&uchGMT, DC_GMT_DIGITS);

    /* extract Serial Number retrieval reference number */
    LIB_cnv_ebcdic_to_ascii ((PUCHAR)(pauchRefNum),
                   (PUCHAR)&uchSerialNum, DC_SERIAL_NUMBER_DIGITS);

    /* null terminate the strings for calls to atol */
    uchGMT[DC_GMT_DIGITS]= '\0';
    uchSerialNum[DC_SERIAL_NUMBER_DIGITS]='\0';

    /* convert GMT to integer and add to machine numbere shifted left by 7
digits */
    ulTemp = uchMachine - START_VENDOR_ADDRESS;
    *pulCICS = ulTemp * DC_SERIAL_NUMBER_SHIFT * DC_MSG_TYPE_SHIFT;
    ulTemp = uchMsgType * DC_SERIAL_NUMBER_SHIFT;
```

```
    *pulCICS += ulTemp;  /* add in msg type digit */
    ulTemp = atol((PCHAR) &uchGMT[0]);
    *pulCICS += ulTemp;  /* add in 7 digits of GMT */

    /* save 3 bytes of serial number into Info array. Note no loss of data since
       the serial number is only 5 digits of value (max 99999) */
    ulTemp = atol((PCHAR) &uchSerialNum[0]);
    memcpy (pauchInfo, &ulTemp, DSM_INFO_SIZE);

    return;
}


/*****************************************************************************
* Module:   DC_Create_RefNum
* Desc:     Create RETRIEVAL_REF_NUMBER from lookup index and info field used by
the DSM in
*           the SHDBCRD thread.
*           Use lower 7 digits from the CICS field (GMT) and 5 digit seerial
number from the
*           Info field.
*
* Inputs:   pauchInfo       Pointer to 3 byte string for machine serial number
(binary)
*           ulCICS          Pointer to long for unique key
*                           The CICS key is:
*                                v,vcg,ggg,ggg
*                           where   'vv' is the vendor number (0-15)
*                                   'c'  is in the range 1 to 6 (the x from EUCx
message type)
*                                   'ggggggg' is the 7 GMT digits from the
RETRIEVAL_REF_NUMBER
* Outputs:  pauchRefNum     Pointer to EBCDIC string for  RETRIEVAL_REF_NUMBER
*           puchMachine     Pointer to machine number;
*
* Errors:   Default Error Processing.
*****************************************************************************/

VOID    DC_Create_RefNum (PUCHAR pauchInfo, ULONG ulCICS, PUCHAR pauchRefNum,
PUCHAR puchMachine)
{


    ULONG   ulSerialNum;
    ULONG   ulGMT;
    ULONG   ulMachine;
    ULONG   ulMsgType;
    PUCHAR  ptr;
    UCHAR   achTempChar[DC_GMT_DIGITS_IN_LONG];

    ptr = (PUCHAR) &ulMachine;

    ulMachine = (ulCICS / DC_SERIAL_NUMBER_SHIFT) / DC_MSG_TYPE_SHIFT;
    ulMsgType = (ulCICS / DC_SERIAL_NUMBER_SHIFT) - (ulMachine *
DC_MSG_TYPE_SHIFT);
    ulGMT = ulCICS - (ulMachine * DC_SERIAL_NUMBER_SHIFT * DC_MSG_TYPE_SHIFT) -
                    (ulMsgType * DC_SERIAL_NUMBER_SHIFT);
```

```
    ulSerialNum = 0;
    memcpy ((CHAR*) &ulSerialNum, pauchInfo, DSM_INFO_SIZE);


    /* move machine number into return parameter */
    *puchMachine  = *ptr + START_VENDOR_ADDRESS;


    /* build Retrieval Reference Number */
    sprintf ((PCHAR) &achTempChar[0], "%*i", DC_SERIAL_NUMBER_DIGITS,
ulSerialNum);
    memcpy(pauchRefNum, &achTempChar, DC_SERIAL_NUMBER_DIGITS);

    /* overlay msg type onto 1st byte since EUC4 and EUC5  Retrieval Reference
Numbers
       are the same */
    /********************** no, this is not implemented in EV !!
    sprintf ((PCHAR) &achTempChar[0], "%*i", 1, ulMsgType);
    *pauchRefNum = *achTempChar;
    *************************/


    sprintf ((PCHAR) &achTempChar[0], "%0*li", DC_GMT_DIGITS, ulGMT);
    ptr = ((UCHAR *)pauchRefNum) +  DC_SERIAL_NUMBER_DIGITS;
    memcpy(ptr, &achTempChar[0], DC_GMT_DIGITS);

   /* convert retrieval reference number to EBCDIC */
    LIB_cnv_ascii_to_ebcdic (pauchRefNum, pauchRefNum,
RETRIEVAL_REF_NUMBER_MAX);




    return;
}


/*****************************************************************************
* Module:   DBCRDT_SendToDevice
* Desc:
* Inputs:   Pointer to message,
*           Message Length,
*           Pointer to Info used to ID the message or device (generated in call
to
*               DBCRDT_SendToHost.
* Outputs:  successful/unsuccessful
* Errors:   Default Error Processing.
*****************************************************************************/
BOOL    DBCRDT_SendToDevice(PUCHAR pszMsg, USHORT usMsgLength, PUCHAR pauchInfo)
{


    APIRET  error_ind;
    INT     iVer;
    INT     iEUC2;
    INT     iBusy;
    UCHAR   szMsgType[MSG_ID_TYPE_MAX];
```

```
    UCHAR   uchStrIndex;


    DC_EXTERNAL_EUC_MSG     tEUC;    /* Mail message to be sent to DC Thread for
EUC2*/
    DC_DEL_TRANS            tDEL;    /* Mail message to be sent to DC Thread for
MACK */
    MSG_C_MACK              *ptMACK;
    char szFepMsg[24];

    iVer = TRUE; /*Assume no errors */
    iEUC2 = FALSE;
    iBusy = FALSE;

    if ((pauchInfo == (PUCHAR) NULL) || (pszMsg == (PUCHAR) NULL))
    {
        iVer = FALSE;
    }
    else if (usMsgLength > sizeof(tEUC))
    {
        iVer = FALSE;
    }

    if (iVer)
    {
        /*
        sprintf (&szFepMsg[0], "To device: " );
        Write_Log_Msg("dctran.hex", szFepMsg,pszMsg,usMsgLength );
        */

        /*ptEUC2 = (DC_EXTERNAL_EUC2_MSG *) pszMsg;*/
        ptMACK = (MSG_C_MACK *) pszMsg;

        tEUC.usSize = usMsgLength;
        /* Copy the message */
        memcpy (&tEUC.tHDR, pszMsg, usMsgLength);

        /* get message code */
        LIB_cnv_ebcdic_to_ascii((PUCHAR) pszMsg,(PUCHAR) &szMsgType,
                                        MSG_ID_TYPE_MAX);
        if (!strncmp(szMsgType, MSG_ID_EUC2, MSG_ID_TYPE_MAX))
        {
            /* for EUC2 the machine number is from the auchInfo field (SCP)*/
            tEUC.uchMachine = pauchInfo[2];
            iEUC2 = TRUE;
        }
        else if (!strncmp(szMsgType, MSG_ID_EUB2, MSG_ID_TYPE_MAX))
        {
            /* for EUC2 the machine number is from the auchInfo field (SCP)*/
            tEUC.uchMachine = pauchInfo[2];
            iEUC2 = TRUE;
        }
        else if (!strncmp(szMsgType, MSG_ID_EUC0, MSG_ID_TYPE_MAX))
        {
            error_ind = WriteToQ (&ulDEB_CRED_handle, SMADS_sel, DC,
                                  (PVOID) NULL, DC_MSG, DC_EUC0_RSP, 0,0);
            if (error_ind != 0)
```

```
            {
               SH_REPORT_EVENT (WARNING_EVENT, SH_STATUS_SWERR,
                                SH_STATUS_IGNORE, error_ind,
                                "Error Returned by WriteToQ");
            }
            iVer = FALSE;
            iEUC2 = FALSE;
        }
        else
        {
            /* If this is a MACK then the machine number is encoded in the CICS
*/

            /* Convert message to DC_DELETE_TRANS delete_msg */
            tDEL.ulCICS = ptMACK->cics_trans_no.v;
            DC_Create_RefNum ((PUCHAR)  pauchInfo,tDEL.ulCICS,
                        (PUCHAR) &tDEL.auchRefNum, (PUCHAR) &tEUC.uchMachine);

            tDEL.uchMachine = tEUC.uchMachine;
            memcpy (&tDEL.uchMackType[0], (PUCHAR) (pszMsg+MSG_ID_TYPE_MAX),
                    MSG_ID_TYPE_MAX);
        }
        if ((tEUC.uchMachine < START_VENDOR_ADDRESS) ||
           (tEUC.uchMachine > (START_VENDOR_ADDRESS+MAX_NUM_VENDORS)))
        {
            if (iVer)
            {
               /* invalid message */
               SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                                   SH_STATUS_IGNORE, (ULONG) tEUC.uchMachine,
                                   "SendToDevice Invalid Machine Number");
               iVer = FALSE;
            }
        }
    }

    if (iVer)
    {
        /* check if machine is talking to us  */
        error_ind = STATCommInitVer(tEUC.uchMachine,&iVer);
        if (iVer && iEUC2)
        {
            uchStrIndex = tEUC.uchMachine  - START_VENDOR_ADDRESS;
            /* check machine is in the correct state to process EUC2 */
            if (tDevice_Str[uchStrIndex].usState[DC_EUC2_EVENT] != DC_IDLE)
            {
                /* Set flag so that we do not send a NAK */
                /* If we are in the wrong state, do not send a response to CC,
                   Let the CC timeout or receive the response for the EUC2
                   that is already inprogress */
                iBusy = TRUE;
                iVer = FALSE;
            }
        }
    }

    if(iVer)
```

```
    {
        if (iEUC2)
        {
            /* euc2 */

            /* send message to DCThread*/
            error_ind = WriteToQ (&ulDEB_CRED_handle, SMADS_sel, DC,
                        &tEUC, DC_MSG, DC_EUC2,
                        usMsgLength + sizeof(tEUC.usSize) +
sizeof(tEUC.uchMachine), 0);
            if (error_ind != 0)
            {
               SH_REPORT_EVENT (WARNING_EVENT, SH_STATUS_SWERR,
                                SH_STATUS_IGNORE, error_ind,
                                "Error Returned by WriteToQ");
                iVer = FALSE;
            }
        }
        if (!strncmp(szMsgType, MSG_ID_MACK, MSG_ID_TYPE_MAX))
        {
            /* Send message to DCThread*/
            error_ind = WriteToQ (&ulDEB_CRED_handle, SMADS_sel, DC,
                        &tDEL, DC_MSG, DC_DELETE_TRANS,
                        sizeof(tDEL) , 0);
            if (error_ind != 0)
            {
               SH_REPORT_EVENT (WARNING_EVENT, SH_STATUS_SWERR,
                                SH_STATUS_IGNORE, error_ind,
                                "Error Returned by WriteToQ");
            }

            /* Set return to false so that the DSM module will not update the
MACK
               message as completed. After the device ACKs this message in the
DC thread,
               then the DC thread will notify the DSM that is is OK to mark the
message
               as completed */
            iVer = FALSE;
        }
    }
    else
    {
        /* Problem with this device, still recieving a previous EUC2)*/
        if ((iEUC2) && (iBusy))
        {
            /* If this is an EUC2 and the device is not in the correct state to
send
               an EUC2, then do nothing. Send TRUE back to shdebcre module so
that
               a NAK is not sent to the CC, but do not send anything to the
device,
               this may be a case where a duplicate EUC2 has come down. Let the
CC wait
               until tthe device responds, or let the CC timeout and reverse the
EUC2.*/
```

```
            /* Send TRUE response back so the EUC2 is not NAKed by SHDEBCRE
module */
            iVer = TRUE;
        }
    }

    return (iVer);

}

/*
*********************************************************************
* Function:         DC_Device
* Desc:             Checks if device is a debit/creit device.
* Inputs:           Device Id
* Outputs:          None
* Return Value:     returns TRUE id device has debiit/credit; else returns FALSE
* External Effects: None
*********************************************************************
*/
INT DC_Device ( UCHAR uchDevice)
{
    APIRET  error_ind;
    int iReturn = 0;
    UCHAR status3;

    error_ind = STATGet (uchDevice, NULL, NULL, &status3, NULL, NULL, NULL);

    if ( status3 &  VEND_EXP_VEND ) /*This is an Express Vendor*/
    {
            iReturn = 1;
    }
    error_ind = error_ind;
    return (iReturn);
}



/*************************************************************************
* Module:   DC_SendTranReq
* Desc:     Send a TX_DC_TRANS_REQ message to a device to kick off the
*           transaction request process for a new transaction.
*
* Inputs:   ptDevice_Str        Pointer to DC_DEVICE_STR for this machin e
*           uchMachine          Machine number
*           ulCurrentTime       Current timestamp
*
* Outputs:  NONE, but ptDevice_Str is updated with new message sequence number
and block
*           and state values.
*
*
* Errors:   Default Error Processing.
*************************************************************************/

VOID    DC_SendTranReq (DC_DEVICE_STR *  ptDevice_Str,
```

```
                        UCHAR uchMachine, ULONG ulCurrentTime)
{

    FEP_MSG_HEADER          DC_Request_Msg;
    ULONG                   ulStatus;
    INT                     iVer;


        memset (&DC_Request_Msg, 0, sizeof(DC_Request_Msg));
        DC_Request_Msg.uchBlock = 0;
        ptDevice_Str->uchBlock = 0;
        if (ptDevice_Str->usSeqNum == USHRT_MAX)
        {
            ptDevice_Str->usSeqNum = 0;
        }
        else
        {
            ++ptDevice_Str->usSeqNum;
        }
        ulStatus = UtOrderTrans((PVOID) &ptDevice_Str->usSeqNum,
                                (PVOID)&DC_Request_Msg.usSeqNum,
                                sizeof (short), sizeof (short));


        /* request next DC transaction from the device */
        ulStatus = STATCommInitVer(uchMachine,&iVer);
        if(iVer)
        {
            ulStatus = FEPMessage (0, (UINT) uchMachine, TX_DC_TRAN,
                                   sizeof( DC_Request_Msg),
                                   &DC_Request_Msg, 0);

            if (ulStatus != 0)
            {
                SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                                 SH_STATUS_IGNORE, ulStatus,
                                 "Error Returned by FEPMessage");
            }
        }
        ptDevice_Str->usRetryCount[DC_SECURE_EVENT] = 0;
        ptDevice_Str->usTryAgain = FALSE;
        ptDevice_Str->usState[DC_SECURE_EVENT] = DC_TRANS_ACK;
        ptDevice_Str->ulTime[DC_SECURE_EVENT] =  ulCurrentTime;
        ptDevice_Str->usTransIndex = 0; /* Start at zero offset */
return;
}


/*************************************************************************
* Module:    DC_CheckNAK
* Desc:      Check if the ACK message is a NAK based upon return code in ACK msg.
*
*
* Inputs:    USHORT usMsgSubcode         Message code for message sent to this
thread.
*            PUCHAR puchData             Pointer to message data sent to this
thread.
```

```
*
* Outputs:  Returns the message ID to process (internal msg id to this module)
*           puchMachine                 Machine number
*           puchStrIndex                Index for this device into global
tDevice_Str
*
*
* Errors:   Default Error Processing.
*****************************************************************************/

USHORT    DC_CheckNAK (PUCHAR puchData, USHORT usMsgSubcode, PUCHAR
puchStrIndex, PUCHAR puchMachine )
{
    USHORT                  usProcMsgId;
    USHORT                  usSeqNum= 0;
    DC_SECURE_DEL_RESPONSE* ptDC_ACK_Msg;
    DC_TRANS_RESPONSE*      ptDC_Trans_Msg;
    ULONG                   ulStatus;

        usProcMsgId = usMsgSubcode;
        switch (usMsgSubcode)
        {
            /* special handling to determine  NAKs from  ACKs */
            case DC_DELETE_ACK:
                ptDC_ACK_Msg = (DC_SECURE_DEL_RESPONSE *) puchData;
                *puchMachine = ptDC_ACK_Msg->uchMachine;
                if ((*puchMachine > MAX_VENDOR_ADDRESS) || (*puchMachine <
START_VENDOR_ADDRESS))
                {
                    SH_REPORT_EVENT(ETDebugOutput,
                    MC_STATUS_NONE, MC_STATUS_IGNORE,
                    (APIRET) *puchMachine,
                     "SMADDEB...invalid machine id in ACK message.");
                    usProcMsgId = 0;
                }
                else
                {
                    *puchStrIndex = *puchMachine - START_VENDOR_ADDRESS;
                    if (ptDC_ACK_Msg->uchResultCode == 1)
                    {
                        usProcMsgId = DC_DELETE_NAK;
                    }
                    /* Return code 2 means nothing to delete, handle as an ACK
                       to send the DSM a message to clean up. */
                    /* Else if return code is 0, check that it matches the
DELETE
                       message that was sent */
                    else if (memcmp (&tDevice_Str[*puchStrIndex].uchDeleteRefNo,
                                     &ptDC_ACK_Msg->uchRefNo,
                                     RETRIEVAL_REF_NUMBER_MAX) !=0)
                    {
                        /* If reference number does not match, then treat as a
NAK */
                        usProcMsgId = DC_DELETE_NAK;
                    }

                    if((ulDC_Debug) && (usProcMsgId == DC_DELETE_NAK))
```

```
                    {
                        sprintf (auchTmpMsg,
                                "SMADDEB...DC DELETE NAK rcvd from device %2i.",
                                *puchMachine);
                        SH_REPORT_EVENT(ETDebugOutput,
                             MC_STATUS_NONE, MC_STATUS_IGNORE,
                             (APIRET) 0, auchTmpMsg);
                    }
                }
                break;
            case DC_SECURE:
                ptDC_ACK_Msg = (DC_SECURE_DEL_RESPONSE *) puchData;
                *puchMachine = ptDC_ACK_Msg->uchMachine;
                if ((*puchMachine > MAX_VENDOR_ADDRESS) || (*puchMachine <
START_VENDOR_ADDRESS))
                {
                    SH_REPORT_EVENT(ETDebugOutput,
                    MC_STATUS_NONE, MC_STATUS_IGNORE,
                    (APIRET) *puchMachine,
                     "SMADDEB...invalid machine id in ACK message.");
                    usProcMsgId = 0;
                }
                else
                {
                    *puchStrIndex = *puchMachine - START_VENDOR_ADDRESS;
                    if (ptDC_ACK_Msg->uchResultCode != 0)
                    {
                        usProcMsgId = DC_SECURE_NAK;
                    }
                    else if (memcmp (&tDevice_Str[*puchStrIndex].uchSecureRefNo,
                                &ptDC_ACK_Msg->uchRefNo,
                                RETRIEVAL_REF_NUMBER_MAX) !=0)
                    {
                        /* If reference number does not match, then treat as a
NAK */
                        usProcMsgId = DC_SECURE_NAK;
                    }
                    if((ulDC_Debug) && (usProcMsgId == DC_SECURE_NAK))
                    {
                        sprintf (auchTmpMsg,
                                "SMADDEB...DC SECURE NAK rcvd from device %2i.",
                                *puchMachine);
                        SH_REPORT_EVENT(ETDebugOutput,
                             MC_STATUS_NONE, MC_STATUS_IGNORE,
                             (APIRET) 0, auchTmpMsg);
                    }
                }
                break;
            case DC_EUC2_ACK:

                ptDC_Trans_Msg = (DC_TRANS_RESPONSE *) puchData;
                *puchMachine = ptDC_Trans_Msg->uchMachine;
                if ((*puchMachine > MAX_VENDOR_ADDRESS) || (*puchMachine <
START_VENDOR_ADDRESS))
                {
                    SH_REPORT_EVENT(ETDebugOutput,
                    MC_STATUS_NONE, MC_STATUS_IGNORE,
```

```
                    (APIRET) *puchMachine,
                     "SMADDEB...invalid machine id in ACK message.");
                    usProcMsgId = 0;
                }
                else
                {

                    *puchStrIndex = *puchMachine - START_VENDOR_ADDRESS;
                    ulStatus = UtOrderTrans((PVOID)&ptDC_Trans_Msg-
>EUC_Msg.tHDR.usSeqNum,
                                            (PVOID)&usSeqNum,
                                            sizeof (short), sizeof (short));
                    /* Check result code in first byte of data protion of
message */
                    if (ptDC_Trans_Msg->EUC_Msg.auchData[0] != 0)
                    {
                        usProcMsgId = DC_EUC2_NAK;
                        sprintf (auchTmpMsg,
                                "SMADDEB...DC EUC2 NAK received from device
%2i.",
                                *puchMachine);
                        SH_REPORT_EVENT(ETDebugOutput,
                             MC_STATUS_NONE, MC_STATUS_IGNORE,
                             (APIRET) 0, auchTmpMsg);
                    }
                    else if ((ptDC_Trans_Msg->EUC_Msg.tHDR.uchBlock ==
                                    (tDevice_Str[*puchStrIndex].uchC2Block-1))
                                    &&
                            (usSeqNum == tDevice_Str[*puchStrIndex].usC2SeqNum))
                    {
                            /* This is a duplicate of the last response
received, just ignore it */
                        usProcMsgId = 0;
                        sprintf (auchTmpMsg,
                            "SMADDEB..EUC2 ACK duplicate block ignored
mach:%2i;block:%2i;expected:%2i",
                             *puchMachine, ptDC_Trans_Msg-
>EUC_Msg.tHDR.uchBlock,
                             tDevice_Str[*puchStrIndex].uchC2Block);
                        SH_REPORT_EVENT(ETDebugOutput,
                             MC_STATUS_NONE, MC_STATUS_IGNORE,
                             (APIRET) 0, auchTmpMsg);

                    }
                    else if (ptDC_Trans_Msg->EUC_Msg.tHDR.uchBlock !=
                                    tDevice_Str[*puchStrIndex].uchC2Block)
                    {
                        /* If block/sequence number does not match, then treat
as a NAK */
                        usProcMsgId = DC_EUC2_NAK;
                        sprintf (auchTmpMsg,
                            "SMADDEB..EUC2 ACK block mismatch;
mach:%2i;block:%2i;expected:%2i",
                             *puchMachine, ptDC_Trans_Msg-
>EUC_Msg.tHDR.uchBlock,
                             tDevice_Str[*puchStrIndex].uchC2Block);
                        SH_REPORT_EVENT(ETDebugOutput,
```

```
                            MC_STATUS_NONE, MC_STATUS_IGNORE,
                            (APIRET) 0, auchTmpMsg);
                    }
                    else if (usSeqNum != tDevice_Str[*puchStrIndex].usC2SeqNum)
                    {
                        /* If block/sequence number does not match, then treat
as a NAK */
                        usProcMsgId = DC_EUC2_NAK;
                        sprintf (auchTmpMsg,
                            "SMADDEB..EUC2 ACK seq num mismatch; mach:%2i;
num:%2i; expected:%2i",
                                *puchMachine,usSeqNum,
tDevice_Str[*puchStrIndex].usC2SeqNum);
                        SH_REPORT_EVENT(ETDebugOutput,
                             MC_STATUS_NONE, MC_STATUS_IGNORE,
                             (APIRET) 0, auchTmpMsg);
                    }
                }
                break;

            default:
                ulStatus = ulStatus; /*get rid of compiler msg */
                break;
        }

        return usProcMsgId;
}

/*****************************************************************************
* Module:   DC_CheckTimeouts
* Desc:     Check if any device has timed out waiting for a response to a
*           last message sent out. This function will retrun after it finds
*           the next device that is waiting.
*
*
* Inputs:   uchMachineIndex        Start index to begin search for device
waiting.
*           uchEventIndex          Type of event watingin on (SECURE, DELETE,
EUC2)
*           ulCurrentTime          Time
*
* Outputs:  Returns TRUE/FALSE     TRUE if no device found
*           usProcMsgId            Message code local to this module if device
found
*           ulWaitTime             Set time to wait on next UTREADQ call
*           uchStrIndex            Index into tDevice_Str for machine waiting
*           uchMachine             machine number in wait state
*
*
* Errors:   Default Error Processing.
*****************************************************************************/
USHORT  DC_CheckTimeouts(UCHAR uchMachineIndex,  UCHAR  uchEventIndex,
                         PUSHORT pusProcMsgId, ULONG ulCurrentTime,
                         PULONG pulWaitTime,
                         PUCHAR puchStrIndex,  PUCHAR puchMachine)
{
    ULONG   ulTempTime;
```

```
    USHORT  usCheck_for_timeouts = TRUE;
    ULONG   ulWaitTime;


    if (uchEventIndex == DC_EUC2_EVENT)
    {
        ulWaitTime = DC_EUC2_MAX_WAIT/1000;
    }
    else
    {
        ulWaitTime = DC_MAX_WAIT/1000;
    }
    if (tDevice_Str[uchMachineIndex].ulTime[uchEventIndex] != ULONG_MAX)
    {
        if (tDevice_Str[uchMachineIndex].ulTime[uchEventIndex] + ulWaitTime <=
ulCurrentTime)
        {
            /* found a device that timed out */
            usCheck_for_timeouts = FALSE; /* exit loop checking for device
timeouts*/
            *puchMachine = uchMachineIndex + START_VENDOR_ADDRESS;
            *puchStrIndex = uchMachineIndex;

            if (tDevice_Str[*puchStrIndex].usState[uchEventIndex] ==
DC_WAITING_ACK)
            {
                if (uchEventIndex == DC_DELETE_EVENT)
                {
                    *pusProcMsgId = DC_DELETE_TIMEOUT;
                }
               else if  (uchEventIndex == DC_SECURE_EVENT)
                {
                    *pusProcMsgId = DC_SECURE_TIMEOUT;
                }
                else if  (uchEventIndex == DC_RESEND_EVENT)
                {
                    *pusProcMsgId = DC_RESEND_TIMEOUT;
                }
            }
            else if (tDevice_Str[*puchStrIndex].usState[uchEventIndex] ==
DC_TRANS_ACK)
            {
                if  (uchEventIndex == DC_SECURE_EVENT)
                {
                    *pusProcMsgId = DC_TRANS_TIMEOUT;
                }
                else if  (uchEventIndex == DC_EUC2_EVENT)
                {
                    *pusProcMsgId = DC_EUC2_TIMEOUT;
                }
            }
        }
        else
        {
            /* A device is still waiting for a response  but has not timed out
yet.
               Need to set time to wait on next UTREADQ call */
```

```
            /* Note that ulWaitTime and MAX_WAIT are in milleseconds, ulTime is
in seconds*/
            ulTempTime =
                ulCurrentTime -
tDevice_Str[uchMachineIndex].ulTime[uchEventIndex];
            if (ulTempTime == 0)
            {
                ulTempTime = ulWaitTime * 1000;
            }
            else
            {
                ulTempTime = ulTempTime * 1000;
            }

            *pulWaitTime =
                ((*pulWaitTime) < ulTempTime) ? (*pulWaitTime) : ulTempTime;
        }
    }
    return usCheck_for_timeouts;
}

/**************************************************************************
* Module:   DC_ProcTranTimeout
* Desc:     Process a timeout waiting for a transaction from a machine.
*
*
* Inputs:   uchStrIndex         Index  into DEVICE_STR for this machine
*           uchMachine          Machine number
*           ulCurrentTime       Current timestamp
*
* Outputs:  pusMsgSent   set to TRUE if a message is sent out. This is used to
determine
*                        how long to wait on next DosReadQueue
*
*
* Errors:   Default Error Processing.
**************************************************************************/

VOID    DC_ProcTranTimeout  (UCHAR uchMachine, UCHAR uchStrIndex,
                             ULONG ulCurrentTime, PUSHORT pusMsgSent)
{
    ULONG                   ulStatus;
    FEP_MSG_HEADER          DC_Request_Msg;
    INT                     iVer;

    if (tDevice_Str[uchStrIndex].usState[DC_SECURE_EVENT] == DC_TRANS_ACK)
    {
        /* Resend if we have not already sent this 3 times */
        if (tDevice_Str[uchStrIndex].usRetryCount[DC_SECURE_EVENT] <
DC_MAX_RETRY)
        {
            ++tDevice_Str[uchStrIndex].usRetryCount[DC_SECURE_EVENT];
            /* Just request a transaction again (start over at block zero) */
            tDevice_Str[uchStrIndex].usTransIndex = 0;
            tDevice_Str[uchStrIndex].uchBlock = 0;
            DC_Request_Msg.uchBlock = tDevice_Str[uchStrIndex].uchBlock;
            ulStatus = UtOrderTrans((PVOID)&tDevice_Str[uchStrIndex].usSeqNum,
```

```
                                    (PVOID)&DC_Request_Msg.usSeqNum,
                                    sizeof (short), sizeof (short));

            /* request next DC transaction from the device */
            ulStatus = STATCommInitVer(uchMachine,&iVer);
            if(iVer)
            {
                ulStatus = FEPMessage (0, (UINT) uchMachine, TX_DC_TRAN,
                                    sizeof( DC_Request_Msg),
                                    &DC_Request_Msg, 0);
                if (ulStatus != 0)
                {
                    SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                                    SH_STATUS_IGNORE, ulStatus,
                                    "Error Returned by FEPMessage");
                }
                else
                {
                    sprintf (auchTmpMsg,
                            "SMADDEB...Resend Request for DC Trans, device
%2i.",
                            uchMachine);
                    SH_REPORT_EVENT(ETDebugOutput,
                         MC_STATUS_NONE, MC_STATUS_IGNORE,
                         (APIRET) 0, auchTmpMsg);
                }
            }
            tDevice_Str[uchStrIndex].usTryAgain = TRUE;
            tDevice_Str[uchStrIndex].ulTime[DC_SECURE_EVENT] =  ulCurrentTime;
            *pusMsgSent = TRUE;
        }
        else if (tDevice_Str[uchStrIndex].usTryAgain)
        {
            /* Request starting from first block of new message */
            DC_SendTranReq (&tDevice_Str[uchStrIndex],
                uchMachine, ulCurrentTime);
            *pusMsgSent = TRUE;
        }
        else
        {
            /* Clean up and go back to idle state. */
            tDevice_Str[uchStrIndex].usRetryCount[DC_SECURE_EVENT] = 0;
            tDevice_Str[uchStrIndex].ulTime[DC_SECURE_EVENT] = ULONG_MAX;
            tDevice_Str[uchStrIndex].usState[DC_SECURE_EVENT] = DC_IDLE;

            SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                                SH_STATUS_IGNORE, (ULONG) uchMachine,
                                "Timeout Waiting for DC Transaction from
Device");
        }
    }
    else
    {   /* Wrong state, just set flag to check for timeouts. */
        *pusMsgSent = TRUE;
    }

}
```

```
/*****************************************************************************
* Module:   DC_Proc_Trans
* Desc:     Process a Debit/Credit message from device: either request next
block
*           or if this is the last block then send it on to the CC.
*
*
* Inputs:   DC_TRANS_RESPONSE  *ptDC_Trans_Msg  Pointer to Device Response
Message,
*           uchStrIndex,        Index into Device_Str for this machine.
*           usProcMsgId         Message Code (can be ab EUC1 or EUC6)
*           uchMachine          Machine number
*           ulCurrentTime       Current timestamp
*
* Outputs:  NONE, messages are sent to the device or to the CC.
*
*
* Errors:   Default Error Processing.
*****************************************************************************/

VOID    DC_Proc_Trans ( DC_TRANS_RESPONSE  *ptDC_Trans_Msg,
                        UCHAR   uchMachine,
                        UCHAR   uchStrIndex,
                        ULONG   ulCurrentTime,
                        USHORT  usProcMsgId)
{
    USHORT          usCRC;
    USHORT          usSeqNum = 0;
    USHORT          usDataSize;
    USHORT          usTemp;
    ULONG           ulStatus;
    ULONG           ulIndex;
    UCHAR           auchInfo[DSM_INFO_SIZE];
    UCHAR           uchMsgType;
    FEP_MSG_HEADER  DC_Request_Msg;
    MSG_C_HEADER    *ptCC_MsgHdr;
    PUCHAR          pauchRefNum;
    INT             iVer;
    INT             iSecure;
    BOOL            bcardbenefit;



    if (tDevice_Str[uchStrIndex].usState[DC_SECURE_EVENT] != DC_TRANS_ACK)
    {   /* Wrong state to accept a response. */
        /* What happens if we get out of synch, how to get state back to idle */
        if (tDevice_Str[uchStrIndex].ulErrorCount < ULONG_MAX)
        {
            ++tDevice_Str[uchStrIndex].ulErrorCount;
        }
         if(ulDC_Debug)
         {
               SH_REPORT_EVENT(ETDebugOutput,
                MC_STATUS_NONE, MC_STATUS_IGNORE,
                    (APIRET) uchMachine,
```

```
                    "SMADDEB...Trans msg rcvd- internal state error.");
        }
    }
    else    /* Correct state to proceed */
    {
        usCRC = 0;
        /* Get the msg seq num from the header into the correct byte order */
        ulStatus = UtOrderTrans((PVOID)&ptDC_Trans_Msg->EUC_Msg.tHDR.usSeqNum,
                                (PVOID)&usSeqNum, sizeof (short), sizeof
(short));
        /* Get the CRC from the header into the correct byte order */
        ulStatus = UtOrderTrans((PVOID)&ptDC_Trans_Msg->EUC_Msg.tHDR.usCRC,
                                (PVOID)&usCRC, sizeof (short), sizeof (short));
        /* get the size in the correct byte order */
        usDataSize = 0;
        ulStatus = UtOrderTrans((PVOID)&ptDC_Trans_Msg->EUC_Msg.tHDR.usLength,
                            (PVOID)&usDataSize, sizeof (short), sizeof (short));


        if (usDataSize > FEP_MAX_MSG_SIZE)
        {
            usTemp = ~usCRC;
            sprintf (auchTmpMsg,
                    "DC Trans invalid size in header, device %2i.",
                    uchMachine);
            SH_REPORT_EVENT(ETDebugOutput,
                 MC_STATUS_NONE, MC_STATUS_IGNORE,
                 (APIRET) usDataSize, auchTmpMsg);
        }
        else
        {
            usTemp = SMADCalcCrc((PUCHAR)&ptDC_Trans_Msg->EUC_Msg.auchData,
usDataSize);
        }
        /* check that the CRC is correct */
        if  (usCRC !=  usTemp)
        {
            /*if(ulDC_Debug)*/
            {
                /* CRC error */
                SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                                    SH_STATUS_IGNORE, uchMachine,
                                    "CRC Mismatch on DC Trans");
            }
            /* DO nothing and let it time out and send request again */
            if (tDevice_Str[uchStrIndex].ulErrorCount < ULONG_MAX)
            {
                ++tDevice_Str[uchStrIndex].ulErrorCount;
            }
        }
        /* check for correct sequence number and block */
        else if ((usSeqNum != tDevice_Str[uchStrIndex].usSeqNum) ||
             (ptDC_Trans_Msg->EUC_Msg.tHDR.uchBlock !=
tDevice_Str[uchStrIndex].uchBlock))
        {
            /* The block or msg seq is wrong.
               DO nothing and let it time out and send request again */
```

```
            if (tDevice_Str[uchStrIndex].ulErrorCount < ULONG_MAX)
            {
                ++tDevice_Str[uchStrIndex].ulErrorCount;
            }
            if(ulDC_Debug)
            {
               SH_REPORT_EVENT(ETDebugOutput,
                 MC_STATUS_NONE, MC_STATUS_IGNORE,
                    (APIRET)uchMachine,
                       "SMADDEB...DC Trans MSN Error.");
            }
        }
        else
        {
            /* Response is OK , go ahead and process */
            ulIndex = tDevice_Str[uchStrIndex].usTransIndex;
            /* Copy this block of the message into the saved EUC message
structure */
            if (ptDC_Trans_Msg->EUC_Msg.tHDR.uchBlock == 0)
            {
                /* copy the header and the data if this is the first block */
                usDataSize += MSG_C_HEADERSIZE;
                memcpy ( &tDevice_Str[uchStrIndex].uchExt_TRANS[ulIndex],
                         &ptDC_Trans_Msg->EUC_Msg.tHDR,
                         usDataSize);
            }
            else if ((ulIndex + usDataSize <= DC_TRANS_MAX_SIZE))
            {
                /* After the first block only copy the data */
                memcpy ( &tDevice_Str[uchStrIndex].uchExt_TRANS[ulIndex],
                         &ptDC_Trans_Msg->EUC_Msg.auchData,
                         usDataSize);
            }
            ulIndex += usDataSize;
            tDevice_Str[uchStrIndex].usTransIndex = ulIndex;
            ++tDevice_Str[uchStrIndex].uchBlock;

            if (ptDC_Trans_Msg->EUC_Msg.tHDR.uchLast == 0)
            {
                /* Not the last block, go get the next one */
                DC_Request_Msg.uchBlock = tDevice_Str[uchStrIndex].uchBlock;
                ulStatus =
UtOrderTrans((PVOID)&tDevice_Str[uchStrIndex].usSeqNum,
                                        (PVOID)&DC_Request_Msg.usSeqNum,
                                        sizeof (short), sizeof (short));
                /* request next DC transaction from the device */
                ulStatus = STATCommInitVer(uchMachine,&iVer);
                if(iVer)
                {
                    ulStatus = FEPMessage (0, (UINT) uchMachine, TX_DC_TRAN,
                                        sizeof( DC_Request_Msg),
                                        &DC_Request_Msg, 0);
                    if (ulStatus != 0)
                    {
                        SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                                        SH_STATUS_IGNORE, ulStatus,
                                        "Error Returned by FEPMessage");
```

```
                    }
                }
                tDevice_Str[uchStrIndex].usTryAgain = FALSE;
                tDevice_Str[uchStrIndex].ulTime[DC_SECURE_EVENT] =
ulCurrentTime;
            }
            else if (ptDC_Trans_Msg->EUC_Msg.tHDR.uchLast > 0)
            {
                /* check for a invalid message size*/
                if (ulIndex > MSG_MAX_LENGTH)
                {
                    /* some error with sending the transaction */
                    SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                                        SH_STATUS_IGNORE, ulIndex,
                                        "Invalid size for DC transaction.");
                    ulIndex = MSG_MAX_LENGTH;
                }

                /* Send on to SHDBCRDT if this is the last block of the message
*/
                /* Need to update some fields in the EUC header */
                ptCC_MsgHdr = (MSG_C_HEADER *)
&tDevice_Str[uchStrIndex].uchExt_TRANS[0];
                /* Put in the final message size in header */
                ulIndex = ulIndex - MSG_C_HEADERSIZE;
                ulStatus = UtOrderTrans((PVOID)&ulIndex,
                                        (PVOID)&ptCC_MsgHdr->length,
                                        sizeof (short), sizeof (short));

                /* Only one CC comms block needed for EUC messages */
                ptCC_MsgHdr->last_block = 1;
                /* Set the seq num to zero */
                ptCC_MsgHdr->sequence_no.v = 0;
                /* check if this is and EUBx or and EUCx */
                bcardbenefit = (ptCC_MsgHdr->type[MSG_ID_TYPE_MAX-2] ==
DC_EBCDIC_B);
                /* get EUC msg type (1 to 6) */
                uchMsgType = ptCC_MsgHdr->type[MSG_ID_TYPE_MAX-1] -
DC_EBCDIC_ZERO;
                /* start at pointer to 1st byte and add the offset to
                   the RETRIEVAL REFERENCE NUMBER dependant upon which EUC
message
                   type this is */
                pauchRefNum = (PUCHAR) &ptCC_MsgHdr->type[0];



                if (bcardbenefit)
                {
                  switch (uchMsgType)
                  {
                    case 1:
                        pauchRefNum += DC_EUB1_RET_REF_OFFSET;
                        break;
                    case 5:
                        pauchRefNum += DC_EUB5_RET_REF_OFFSET;
                        break;
```

```
                    case 3:
                        pauchRefNum += DC_EUB3_RET_REF_OFFSET;
                        break;
                    default:
                        pauchRefNum = 0;
                        break;
                   }
                }
                else
                {
                  switch (uchMsgType)
                  {
                    case 1:
                    case 6:
                        pauchRefNum += DC_EUC1_RET_REF_OFFSET;
                        break;
                    case 5:
                        pauchRefNum += DC_EUC5_RET_REF_OFFSET;
                        break;
                    case 3:
                    case 4:
                        pauchRefNum += DC_EUC3_RET_REF_OFFSET;
                        break;
                    default:
                        pauchRefNum = 0;
                        break;
                  }
                }
                if (pauchRefNum == 0)
                {
                    if(ulDC_Debug)
                    {
                       SH_REPORT_EVENT(ETDebugOutput,
                         MC_STATUS_NONE, MC_STATUS_IGNORE,
                            (APIRET)uchMachine,
                               "SMADDEB...DC Trans Unknown Message.");
                    }
                }
                else
                {
                    /* build the CICS Trans Number */
                    DC_Create_DSM_Index( pauchRefNum,
                                         uchMsgType,
                                         uchMachine,
                                         (PUCHAR) &auchInfo[0],
                                         (PULONG) &(ptCC_MsgHdr-
>cics_trans_no.v));
                    /* Save the retrieval reference number for subsequent comms
with
                       the device. */
                    memcpy (&tDevice_Str[uchStrIndex].uchSecureRefNo,
                            pauchRefNum,
                            RETRIEVAL_REF_NUMBER_MAX);

                    /* send it to host */
                    ulIndex += MSG_C_HEADERSIZE;
```

```
                    iSecure = DBCRDT_SendToHost((PUCHAR)&ptCC_MsgHdr-
>type[0],ulIndex,
                                                (PUCHAR) &auchInfo);
                    if (!iSecure)
                    {
                        /* some error with sending the transaction to
SHDEBCRED*/
                        if (tDevice_Str[uchStrIndex].ulErrorCount < ULONG_MAX)
                        {
                            ++tDevice_Str[uchStrIndex].ulErrorCount;
                        }
                        /* some error with sending the transaction */
                        SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                                            SH_STATUS_IGNORE, ulStatus,
                                            "DC Thread can not send to host");
                        /*
                            Not much can be done here, just make sure a SECURE
message
                            is not sent to the device and keep requesting
transactions
                            as if everything was OK
                        */
                    }
                    /* Check that device is past the comms init stage */
                    /* then send the DC_SECURE message to let device go on to
next trans. */
                    ulStatus = STATCommInitVer(uchMachine,&iVer);
                    if(iVer)
                    {
                        if ((usProcMsgId == DC_TRANS) && (iSecure))
                        {
                            /* For EUB5, EUC3, EUC4, and EUC5 send
TX_DC_SECURE_REQ */
                            ulStatus = FEPMessage (
                                0, (UINT) uchMachine,
                                TX_DC_SECURE,
                                RETRIEVAL_REF_NUMBER_MAX,
                                &tDevice_Str[uchStrIndex].uchSecureRefNo[0],
0);
                            if (ulStatus != 0)
                            {
                                SH_REPORT_EVENT (   WARNING_EVENT,
SH_STATUS_SWERR,
                                                    SH_STATUS_IGNORE, ulStatus,
                                                    "Error Returned by FEPMessage");
                            }
                            else
                            {
                               if(ulDC_Debug)
                               {
                                    sprintf(auchTmpMsg,"SECURE:" );
                                    /*Write_Log_Msg("dc1.hex", auchTmpMsg,
                                       (PUCHAR)
&tDevice_Str[uchStrIndex].uchSecureRefNo[0],
                                        RETRIEVAL_REF_NUMBER_MAX);*/
                               }
                            }
```

```
                            if (ptDC_Trans_Msg->EUC_Msg.tHDR.uchLast > 1)
                            {
                                /* If more messages, then set flag to request
again */
                                 tDevice_Str[uchStrIndex].usTryAgain = TRUE;
                            }
                            else
                            {
                                 tDevice_Str[uchStrIndex].usTryAgain = FALSE;
                            }

                            /* Change state to waiting for ACK on SECURE msg */
                            tDevice_Str[uchStrIndex].usState[DC_SECURE_EVENT] =
DC_WAITING_ACK;
                            tDevice_Str[uchStrIndex].ulTime[DC_SECURE_EVENT] =
ulCurrentTime;

tDevice_Str[uchStrIndex].usRetryCount[DC_SECURE_EVENT]  = 0;
                        }
                        else if ((ptDC_Trans_Msg->EUC_Msg.tHDR.uchLast > 1) ||
(!iSecure))
                        {
                            /* If more message  bit is set... */
                            /* For EUB1, EUC1 and EUC6 just request the next
transaction. */
                            /* Request starting from first block of new message
*/
                            DC_SendTranReq (&tDevice_Str[uchStrIndex],
uchMachine,
                                            ulCurrentTime);
                        }
                        else
                        {
                             tDevice_Str[uchStrIndex].usTryAgain = FALSE;
                             tDevice_Str[uchStrIndex].usState[DC_SECURE_EVENT] =
DC_IDLE;
                             tDevice_Str[uchStrIndex].ulTime[DC_SECURE_EVENT] =
ULONG_MAX;

tDevice_Str[uchStrIndex].usRetryCount[DC_SECURE_EVENT]  = 0;
                        }
                    }
                }

            }
        }
    }
}

/*****************************************************************************
* Module:   DC_ProcEUC2
* Desc:     Store and forward an EUC2 Message received from CC destined to a
device.
*           Message must be broken into multiple blocks.
*
*
* Inputs:   uchStrIndex         Index into  DEVICE_STR for this machine
```

```
*           uchMachine          Machine number
*           ulCurrentTime       Current timestamp
*
* Outputs:  NONE,
*
*
* Errors:   Default Error Processing.
*****************************************************************************/

VOID   DC_ProcEUC2 (UCHAR uchMachine, UCHAR uchStrIndex,
                    ULONG ulCurrentTime, DC_EXTERNAL_EUC_MSG *ptDC_Ext_EUC_Msg)
{

    INT                     iRemSize;
    INT                     iMsgSize;
    ULONG                   ulStatus;
    USHORT                  usDataSize;
    USHORT                  usCRC;
    TX_MULTI_BLOCK_MSG      DC_EUC_ToDevice;
    BOOL                    bcardbenefit;


    /* EUC2 received from SHDBCRED (SendToDevice) */
    if (tDevice_Str[uchStrIndex].usState[DC_EUC2_EVENT] == DC_IDLE)
    {
        /* start sending the msg */
        tDevice_Str[uchStrIndex].usBytePtr = 0;
        tDevice_Str[uchStrIndex].iOrgSize = ptDC_Ext_EUC_Msg->usSize;
        tDevice_Str[uchStrIndex].iMsgSize = ptDC_Ext_EUC_Msg->usSize;
        ++tDevice_Str[uchStrIndex].usC2SeqNum;
        tDevice_Str[uchStrIndex].usRetryCount[DC_EUC2_EVENT] = 0;

        memcpy (&tDevice_Str[uchStrIndex].EUC2,
                &ptDC_Ext_EUC_Msg->tHDR,
                ptDC_Ext_EUC_Msg->usSize);
        tDevice_Str[uchStrIndex].ulEUC2_CICS =
            ptDC_Ext_EUC_Msg->tHDR.cics_trans_no.v;


        /* Copy the message into this msg block to be sent to device. */
        iRemSize = tDevice_Str[uchStrIndex].iMsgSize;
        iMsgSize =
            ( iRemSize < MAX_MESSAGE_SIZE) ? iRemSize : MAX_MESSAGE_SIZE;
        memcpy (&DC_EUC_ToDevice.tHDR,

&tDevice_Str[uchStrIndex].EUC2[tDevice_Str[uchStrIndex].usBytePtr],
                iMsgSize);
        DC_EUC_ToDevice.tHDR.uchBlock = 0;
        tDevice_Str[uchStrIndex].uchC2Block = 0;
        if (iMsgSize == iRemSize)
        {
            DC_EUC_ToDevice.tHDR.uchLast = 1;
        }
        else
        {
            DC_EUC_ToDevice.tHDR.uchLast = 0;
        }
```

```
        /* Size of data is msg size minus the header size */
        usDataSize = iMsgSize - MSG_C_HEADERSIZE;
        ulStatus = UtOrderTrans (   (PVOID)&usDataSize,
                                    (PVOID)&DC_EUC_ToDevice.tHDR.usLength,
                                    sizeof (short), sizeof (short));

        ulStatus = UtOrderTrans (   (PVOID)&tDevice_Str[uchStrIndex].usC2SeqNum,
                                    (PVOID)&DC_EUC_ToDevice.tHDR.usSeqNum,
                                    sizeof (short), sizeof (short));


        /* Calculate the CRC on the data portion of the message */
        usCRC =  SMADCalcCrc((PUCHAR)&DC_EUC_ToDevice.auchData, usDataSize);
        ulStatus = UtOrderTrans (   (PVOID)&usCRC,
                                    (PVOID)&DC_EUC_ToDevice.tHDR.usCRC,
                                    sizeof (short), sizeof (short));



        /* Send the mesage to the device via the SMADFEP Thread */
        ulStatus = FEPMessage (0, (UINT) uchMachine, TX_DC_AUTHORIZE,
                                iMsgSize,  &DC_EUC_ToDevice, 0);
        if (ulStatus != 0)
        {
            SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                            SH_STATUS_IGNORE, ulStatus,
                            "Error Returned by FEPMessage");
        }
        else
        {
            tDevice_Str[uchStrIndex].usState[DC_EUC2_EVENT] = DC_TRANS_ACK;
            tDevice_Str[uchStrIndex].ulTime[DC_EUC2_EVENT] = ulCurrentTime;
            if(ulDC_Debug)
            {
                  /* check if this is and EUBx or and EUCx */
                  bcardbenefit = (ptDC_Ext_EUC_Msg->tHDR.type[MSG_ID_TYPE_MAX-2]
== DC_EBCDIC_B);
                  if (bcardbenefit)
                  {
                       sprintf (auchTmpMsg,
                        "SMADDEB...EUB2 Block %1i  Sent to Device %2i",
                        ulStatus, uchMachine);
                  }
                  else
                  {
                       sprintf (auchTmpMsg,
                        "SMADDEB...EUC2 Block %1i  Sent to Device %2i",
                        ulStatus, uchMachine);
                  }
                  SH_REPORT_EVENT(ETDebugOutput,
                     MC_STATUS_NONE, MC_STATUS_IGNORE,
                     (APIRET) uchMachine, auchTmpMsg);
            }
        }
    }
    else
```

```
    {
        /* do not send a NAK back to CC, let it timeout */
        if (tDevice_Str[uchStrIndex].ulErrorCount < ULONG_MAX)
        {
            ++tDevice_Str[uchStrIndex].ulErrorCount;
            SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                                SH_STATUS_IGNORE, (ULONG) uchMachine,
                                "Auhtorization Received but DC Thread is busy");
        }
    }
}

/*****************************************************************************
* Module:   DC_ProcEUC2_ACK
* Desc:     Process an ACK of a EUC2 message block sent to the device.
*           If more blocks to send, continue with next block.
*
* Inputs:   uchStrIndex         Index into  DEVICE_STR for this machine
*           uchMachine          Machine number
*           ulCurrentTime       Current timestamp
*
* Outputs:  pusMsgSent  set to TRUE if a message is sent to the device.
*
* Errors:   Default Error Processing.
*****************************************************************************/

VOID    DC_ProcEUC2_ACK (UCHAR uchMachine, UCHAR uchStrIndex,
                         ULONG ulCurrentTime, PUSHORT pusMsgSent)
{

    INT                     iRemSize;
    INT                     iMsgSize;
    ULONG                   ulStatus;
    USHORT                  usDataSize;
    USHORT                  usCRC;
    TX_MULTI_BLOCK_MSG      DC_EUC_ToDevice;
    UCHAR                   auchInfo[DSM_INFO_SIZE];
    MSG_C_MACK              DC_MACK_Msg;
    BOOL                    bcardbenefit;

    /* continue sending the message */
    if (tDevice_Str[uchStrIndex].uchC2Block == 0)
    {
        tDevice_Str[uchStrIndex].usBytePtr += MAX_MESSAGE_SIZE;
        tDevice_Str[uchStrIndex].iMsgSize -= MAX_MESSAGE_SIZE;
    }
    else
    {   /* If not first block, subtract out header size since header is resent
           eith each block */
        tDevice_Str[uchStrIndex].usBytePtr +=
            (MAX_MESSAGE_SIZE-MSG_C_HEADERSIZE);
        tDevice_Str[uchStrIndex].iMsgSize -=
            (MAX_MESSAGE_SIZE-MSG_C_HEADERSIZE);
    }
    ++tDevice_Str[uchStrIndex].uchC2Block;
    /* do not clear retry count, avoid the infinite loop of NAK/ACK/NAK/ACK */
    /*tDevice_Str[uchStrIndex].usRetryCount[DC_EUC2_EVENT] = 0; */
```

```
    if (tDevice_Str[uchStrIndex].iMsgSize > 0)
    {
        /* Send the next block */
        /* Copy the message into this msg block to be sent to device. */
        iRemSize = tDevice_Str[uchStrIndex].iMsgSize + MSG_C_HEADERSIZE;
        iMsgSize =
            ( iRemSize < MAX_MESSAGE_SIZE) ? iRemSize : MAX_MESSAGE_SIZE;
        /* Copy the header portion into all blocks */
        memcpy (&DC_EUC_ToDevice.tHDR,
                &tDevice_Str[uchStrIndex].EUC2[0],
                MSG_C_HEADERSIZE);
        /* Size of data is msg size minus the header size */
        usDataSize = iMsgSize - MSG_C_HEADERSIZE;
        memcpy (&DC_EUC_ToDevice.auchData,

&tDevice_Str[uchStrIndex].EUC2[tDevice_Str[uchStrIndex].usBytePtr],
                usDataSize);
        DC_EUC_ToDevice.tHDR.uchBlock = tDevice_Str[uchStrIndex].uchC2Block;

        if (iMsgSize == iRemSize)
        {
            DC_EUC_ToDevice.tHDR.uchLast = 1;
        }
        else
        {
            DC_EUC_ToDevice.tHDR.uchLast = 0;
        }

        ulStatus = UtOrderTrans (   (PVOID)&usDataSize,
                                    (PVOID)&DC_EUC_ToDevice.tHDR.usLength,
                                    sizeof (short), sizeof (short));

        /* Calculate the CRC on the data portion of the message */
        usCRC =  SMADCalcCrc((PUCHAR)&DC_EUC_ToDevice.auchData, usDataSize);
        ulStatus = UtOrderTrans (   (PVOID)&usCRC,
                                   (PVOID)&DC_EUC_ToDevice.tHDR.usCRC,
                                   sizeof (short), sizeof (short));

        ulStatus = UtOrderTrans ((PVOID)&tDevice_Str[uchStrIndex].usC2SeqNum,
                                   (PVOID)&DC_EUC_ToDevice.tHDR.usSeqNum,
                                   sizeof (short), sizeof (short));



        /* Send the mesage to the device via the SMADFEP Thread */
        ulStatus = FEPMessage (0, (UINT) uchMachine, TX_DC_AUTHORIZE,
                                   iMsgSize,
                                   &DC_EUC_ToDevice, 0);

        if (ulStatus != 0)
        {
            SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                             SH_STATUS_IGNORE, ulStatus,
                             "Error Returned by FEPMessage");
        }
        else
        {
```

```
            tDevice_Str[uchStrIndex].ulTime[DC_EUC2_EVENT] = ulCurrentTime;
            *pusMsgSent = TRUE;
           if(ulDC_Debug)
           {
                 /* check if this is and EUBx or and EUCx */
                 bcardbenefit = (tDevice_Str[uchStrIndex].EUC2[MSG_ID_TYPE_MAX-
2] == DC_EBCDIC_B);
                 if (bcardbenefit)
                 {
                    sprintf (auchTmpMsg,
                         "SMADDEB...EUB2 Block %1i  Sent to Device %2i",
                        (INT) DC_EUC_ToDevice.tHDR.uchBlock, (INT) uchMachine);
                 }
                 else
                 {
                    sprintf (auchTmpMsg,
                         "SMADDEB...EUC2 Block %1i  Sent to Device %2i",
                        (INT) DC_EUC_ToDevice.tHDR.uchBlock, (INT) uchMachine);
                 }
                 SH_REPORT_EVENT(ETDebugOutput,
                    MC_STATUS_NONE, MC_STATUS_IGNORE,
                    (APIRET) uchMachine, auchTmpMsg);
            }
        }
    }
    else
    {
        /* Send a MACK to SHDEBCRD */
        /* EUC2 is recvd at vendor, send a MACK back to the CC */
        /* Add 'EUC2' to mack */
        LIB_cnv_ascii_to_ebcdic (MSG_ID_MACK, (PUCHAR)&DC_MACK_Msg.mack_id,
                                 MSG_ID_TYPE_MAX);
        memcpy (&DC_MACK_Msg.type,
                &tDevice_Str[uchStrIndex].EUC2[0],
                MSG_ID_TYPE_MAX);
        /* Add CICS trans no to MACK */
        DC_MACK_Msg.cics_trans_no.v = tDevice_Str[uchStrIndex].ulEUC2_CICS;
        DC_MACK_Msg.status = MSG_ACKSTS_OK;
        if (DBCRDT_SendToHost((PUCHAR)&DC_MACK_Msg, (ULONG)
MSG_C_MACKSIZE,(PUCHAR)&auchInfo))
        {
            ulDC_Debug = ulDC_Debug;
        }
        else
        {
           SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                               SH_STATUS_IGNORE, (APIRET) 0,
                               "DC Thread can not MACK to host");
        }
        tDevice_Str[uchStrIndex].usState[DC_EUC2_EVENT] = DC_IDLE;
        tDevice_Str[uchStrIndex].usRetryCount[DC_EUC2_EVENT] = 0;
        tDevice_Str[uchStrIndex].ulTime[DC_EUC2_EVENT] = ULONG_MAX;
    }

}

/*****************************************************************************
```

```
* Module:   DC_ProcEUC2_NAK
* Desc:     Process an NAK or timeout of a EUC2 message block sent to the
device.
*           If retry count has not been exceeded, continue with same block.
*
* Inputs:   uchStrIndex         Index into  DEVICE_STR for this machine
*           uchMachine          Machine number
*           ulCurrentTime       Current timestamp
*
* Outputs:  pusMsgSent  set to TRUE if a message is sent to the device.
*
* Errors:   Default Error Processing.
*****************************************************************************/

VOID    DC_ProcEUC2_NAK (UCHAR uchMachine, UCHAR uchStrIndex, USHORT
usProcMsgId,
                         ULONG ulCurrentTime, PUSHORT pusMsgSent)
{

    INT                     iRemSize;
    INT                     iMsgSize;
    INT                     iTmpPtr;
    ULONG                   ulStatus;
    USHORT                  usDataSize;
    USHORT                  usCRC;
    TX_MULTI_BLOCK_MSG      DC_EUC_ToDevice;
    UCHAR                   auchInfo[DSM_INFO_SIZE];
    MSG_C_MACK              DC_MACK_Msg;


    /* continue sending the same block of the message */
    /* resend the same block, increment number of times this block sent */
    if (tDevice_Str[uchStrIndex].usRetryCount[DC_EUC2_EVENT] <
DC_EUC2_MAX_RETRY)
    {

        ++(tDevice_Str[uchStrIndex].usRetryCount[DC_EUC2_EVENT]);

        /* Send the next block */
        /* Copy the header portion */
        memcpy (&DC_EUC_ToDevice.tHDR,
                &tDevice_Str[uchStrIndex].EUC2[0],
                MSG_C_HEADERSIZE);

        iTmpPtr = tDevice_Str[uchStrIndex].usBytePtr;
        if (usProcMsgId == DC_EUC2_NAK)
        {
            /* restart sending at start of message if a NAK */
            tDevice_Str[uchStrIndex].usBytePtr = 0;
            iTmpPtr = MSG_C_HEADERSIZE;
            tDevice_Str[uchStrIndex].uchC2Block = 0;
            tDevice_Str[uchStrIndex].iMsgSize =
tDevice_Str[uchStrIndex].iOrgSize;
        }
        /* Copy the message into this msg block to be sent to device. */
        if (tDevice_Str[uchStrIndex].uchC2Block == 0)
        {
```

```
        iRemSize = tDevice_Str[uchStrIndex].iMsgSize;
    }
    else
    {
        /* If not at block zero, copy the header and data from where
            the last block left off */
        iRemSize = tDevice_Str[uchStrIndex].iMsgSize + MSG_C_HEADERSIZE;
    }

    iMsgSize =
        ( iRemSize < MAX_MESSAGE_SIZE) ? iRemSize : MAX_MESSAGE_SIZE;
    /* Size of data is msg size minus the header size */
    usDataSize = iMsgSize - MSG_C_HEADERSIZE;
    /* Copy the data portion of the message */
    memcpy (&DC_EUC_ToDevice.auchData,
            &tDevice_Str[uchStrIndex].EUC2[iTmpPtr],
            usDataSize);
    if (iMsgSize == iRemSize)
    {
        DC_EUC_ToDevice.tHDR.uchLast = 1;
    }
    else
    {
        DC_EUC_ToDevice.tHDR.uchLast = 0;
    }

    DC_EUC_ToDevice.tHDR.uchBlock = tDevice_Str[uchStrIndex].uchC2Block;
    ulStatus = UtOrderTrans (   (PVOID)&usDataSize,
                                (PVOID)&DC_EUC_ToDevice.tHDR.usLength,
                                sizeof (short), sizeof (short));

    /* Calculate the CRC on the data portion of the message */
    usCRC =  SMADCalcCrc((PUCHAR)&DC_EUC_ToDevice.auchData, usDataSize);
    ulStatus = UtOrderTrans (   (PVOID)&usCRC,
                            (PVOID)&DC_EUC_ToDevice.tHDR.usCRC,
                            sizeof (short), sizeof (short));
    ulStatus = UtOrderTrans ((PVOID)&tDevice_Str[uchStrIndex].usC2SeqNum,
                            (PVOID)&DC_EUC_ToDevice.tHDR.usSeqNum,
                            sizeof (short), sizeof (short));


    /* Send the message to the device via the SMADFEP Thread */
    ulStatus = FEPMessage (0, (UINT) uchMachine, TX_DC_AUTHORIZE,
                            iMsgSize,
                            &DC_EUC_ToDevice, 0);
    if (ulStatus != 0)
    {
        /* Just wait and timeout to try again */
        SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                        SH_STATUS_IGNORE, ulStatus,
                        "Error Returned by FEPMessage");
    }
    else
    {
          sprintf (auchTmpMsg,
                "SMADDEB...Resend Authorization Block %1i  Sent to Device
%2i",
```

```
                    (INT) DC_EUC_ToDevice.tHDR.uchBlock, (INT) uchMachine);
                SH_REPORT_EVENT(ETDebugOutput,
                  MC_STATUS_NONE, MC_STATUS_IGNORE,
                  (APIRET) uchMachine, auchTmpMsg);
        }
        tDevice_Str[uchStrIndex].ulTime[DC_EUC2_EVENT] = ulCurrentTime;
        *pusMsgSent = TRUE;

    }
    else
    {
        /* Time to give up on sending the EUC2 */
        /* Send a MACK to SHDEBCRD */
        /* Add 'EUC2' to mack */
        LIB_cnv_ascii_to_ebcdic (MSG_ID_MACK, (PUCHAR)&DC_MACK_Msg.mack_id,
                                 MSG_ID_TYPE_MAX);
        memcpy (&DC_MACK_Msg.type,
                &tDevice_Str[uchStrIndex].EUC2[0],
                MSG_ID_TYPE_MAX);
        /* Add CICS trans no to MACK */
        DC_MACK_Msg.cics_trans_no.v = tDevice_Str[uchStrIndex].ulEUC2_CICS;
        DC_MACK_Msg.status = MSG_ACKSTS_ABORT;
        if (DBCRDT_SendToHost((PUCHAR)&DC_MACK_Msg, (ULONG)
MSG_C_MACKSIZE,(PUCHAR)&auchInfo))
        {
            if(ulDC_Debug)
            {
               if (tDevice_Str[uchStrIndex].EUC2[MSG_ID_TYPE_MAX-2] ==
DC_EBCDIC_B)
               {
                  SH_REPORT_EVENT(ETDebugOutput,
                  MC_STATUS_NONE, MC_STATUS_IGNORE,
                     (APIRET) 0,
                        "SMADS info...DC EUB2 NAK sent to host");
               }
               else
               {
                  SH_REPORT_EVENT(ETDebugOutput,
                  MC_STATUS_NONE, MC_STATUS_IGNORE,
                     (APIRET) 0,
                        "SMADS info...DC EUC2 NAK sent to host");
               }
            }
        }
        else
        {
           SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                               SH_STATUS_IGNORE, (APIRET) 0,
                               "DC Thread can not send NAK to host");
        }

        tDevice_Str[uchStrIndex].usRetryCount[DC_EUC2_EVENT] = 0;
        tDevice_Str[uchStrIndex].ulTime[DC_EUC2_EVENT] = ULONG_MAX;
        tDevice_Str[uchStrIndex].usState[DC_EUC2_EVENT] = DC_IDLE;
    }
}
```

```
/*****************************************************************************
* Module:   DCThread
* Desc:     Debit/Credit Device Interface Thread
*           Handles all device communcations (via FEP thread) for Debit/Credit
*           messages. This includes :
*               DC_TRANS request (SMADS request to device to send a DC
transaction)
*               DC_TRANS response (EUC3, EUC4, EUC5, EUB3, EUB5 transaction
block from device)
*               DC_AUTHORIZE request (EUB1, EUC1 or EUC6 message block from
device)
*               DC_AUTHORIZE_ACK (ACK from device for EUC2 block)
*               DC_SECURE request (SMADS message to device that DC_TRANS
received OK)
*               DC_SECURE response (Device ACK to DC_SECURE request message)
*               DC_DELETE request (SMADS message to device to auhtorize DC Trans
deleteion)
*               DC_DELETE response (Device ACK to DC_DELETE message)
*               DC_RESEND (SMADS message to device to re-send all available DC
transactions)
*
*           This Thread also receives messages from other SMADS threads
*               DC_EUC2 (EUC2 message from CC via SHDBCRDT module
*                       This results in a DC_AUTHORIZE msg out to device.
*               MACK    (CC confirms receipt of EUC3, EUC4, EUC5 via SHDBCRDT
module)
*                       This results in a DC_DELETE msg out to device.
*               DC_TRANS_GET (message from SMADSTAT thread to initiate getting a
DC transaction.
*
*           This thread will receive EUCx transactions sent from devices.
*           Since these may be in mutliple FEP blocks, this thread will receive
*           each block as it is sent and assemble the complete EUCx transaction.
*           The complete transaction will then be passed on to the SHDBCRDT
module
*           for storage (if required) and forwarding to the CC.
*           This thread will respond to the device with a SECURE DEBIT/CREDIT
TRANSACTION
*           message to acknowledge receipt at the SMADS.
*           This thread will also receive DELETE DEBIT/CREDIT TRANSACTION
messages from the
*           SHDBCRDT_SendToDevice function to inform the device when a
transaction has
*           been received at the CC, so that it can be deleted. These
transactions will be
*           sent to the device and if a response from the device is not received
within
*           the defined timeout period, the message will be re-sent to the
device. Three
*           attempts will be made to send the message before it wil be thrown
out.
*
*           Also, once a day (after opreational hours), this thread will send a
RESEND
*           DEBIT/CREDIT TRANSACTION message to all devices as a safety measure
to make sure
```

```
*           that the device does not have any left over debit credit
transactions that have not
*           been yet deleted. Note that this could result in a duplicate EUCx
transaction
*           being sent to the CC but the CC will resolve this any not process
duplicates.
*
* Inputs:   N/A
* Outputs:  N/A
* Errors:   Default Error Processing.
*****************************************************************************/


VOID ENV_CDECL DCThread(PVOID pvDummy)
{

    APIRET              ulStatus;

    HQUEUE              ulDC_CCTX_Handle;
    HEV                 DC_QRead_Sem;


    USHORT              usQlen;
    USHORT              usMsgCode;
    USHORT              usMsgSubcode;
    USHORT              usProcMsgId;
    USHORT              usMsgRead;
    USHORT              usMsgSent;
    USHORT              usMore_to_do;
    USHORT              usCheck_for_timeouts;
    USHORT              usFreeMsg;
    USHORT              usRunning;
    PUCHAR              puchData;

    UCHAR               uchMachine;
    UCHAR               uchStart;
    UCHAR               uchEnd;
    UCHAR               uchMachineIndex;
    UCHAR               uchEventIndex;
    UCHAR               uchStrIndex;
    UCHAR               auchInfo[DSM_INFO_SIZE];

    INT                 iQpri;
    INT                 iINDX1;
    INT                 iINDX2;
    INT                 iVer;
    BOOL                bEUC0_Pending = FALSE;

    ULONG               ulWaitTime;
    ULONG               ulPid;
    ULONG               ulEUC0_Time=0;

    TIME_T              ulResendTime;
    TIME_T              ulCurrentTime;
    TIME_T              ulTempTime;

    DC_SECURE_DEL_RESPONSE  *ptDC_ACK_Msg;
```

```
DC_TRANS_RESPONSE       *ptDC_Trans_Msg;
MSG_C_HEADER            EUC0_Msg;
DC_DEL_TRANS            *ptDC_DelTrans;
MSG_C_MACK              DC_MACK_Msg;
DC_EXTERNAL_EUC_MSG     *ptDC_Ext_EUC_Msg;
struct  msg_str         *ptMsg;

REQUESTDATA             qdata;



ulStatus = DosCreateEventSem( NULL, &DC_QRead_Sem, 0L, TRUE );
if (ulStatus != 0)
{
    usRunning = FALSE;
    SH_REPORT_EVENT(FATAL_EVENT, SH_STATUS_SWERR,
        SH_STATUS_IGNORE, ulStatus,
        "Error Returned by DosCreateEventSem");
}


/* Bogus assignment to get rid of compile warning */
/* Actual assignment takes place in UtReadQ */
ptMsg = (struct msg_str *) &ulPid;
usRunning = TRUE;
ulStatus = UtReadQ (ulDEB_CRED_handle,
                    &qdata,
                    &usQlen,
                    &usMsgCode ,
                    &usMsgSubcode,
                    &puchData,
                    NULL,
                    (PVOID) &ptMsg,
                    DC,
                    &iQpri,
                    DCWW_NOWAIT,
                    &DC_QRead_Sem,
                    0);

if (ulStatus != 0)
{
       if (ulStatus != ERROR_QUE_EMPTY)
       {
           usRunning = FALSE;
           SH_REPORT_EVENT(FATAL_EVENT, SH_STATUS_SWERR,
               SH_STATUS_IGNORE, ulStatus,
               "Error Returned by UtReadQ");
       }
}

/* Open a queue for sending messages to SMAD CCTX Thread */
do
{
    ulStatus = DosOpenQueue (&ulPid, &ulDC_CCTX_Handle, CCTX_QUEUE_NAME);
    if (ulStatus != 0)
    {
        DosSleep (1000L);
```

```
        }
    }
    while (ulStatus != 0);


    memset (&tDevice_Str, 0, sizeof(tDevice_Str));

    /* get rid of compiler warning */
    puchData = (PUCHAR) pvDummy;


    for (iINDX1 = 0; iINDX1 < MAX_NUM_VENDORS; ++iINDX1)
    {
        for (iINDX2 = 0; iINDX2 < DC_MAX_EVENT; ++iINDX2)
        {
            tDevice_Str[iINDX1].usState[iINDX2] = DC_IDLE;
            tDevice_Str[iINDX1].ulTime[iINDX2] = ULONG_MAX;
        }
    }

    /* Set up MACK and EUC0 message */
    LIB_cnv_ascii_to_ebcdic (MSG_ID_MACK,
                             (PUCHAR)&DC_MACK_Msg.mack_id,MSG_ID_TYPE_MAX);
    LIB_cnv_ascii_to_ebcdic (MSG_ID_EUC0,
                             (PUCHAR)&EUC0_Msg.type,MSG_ID_TYPE_MAX);
    EUC0_Msg.trans_rev_lvl = LIB_MSG_EUC0_REV;      /* transactn. revision level
*/
    EUC0_Msg.last_block = 1;            /* last msg. block specfier */
    EUC0_Msg.cics_trans_no.v = 0;       /* msg. id of sending task */
    EUC0_Msg.sequence_no.v = 0;     /* message block/sequence no */
    EUC0_Msg.length.v = 0;              /* length of data block */



    /* Get time for next RESEND message to devices (3AM on the next day) */
    ulTempTime = SHTIM_time();
    ulResendTime = ((ulTempTime / SECONDS_PER_DAY) * SECONDS_PER_DAY) +
SECONDS_PER_DAY;
    ulResendTime += SIX_HOURS;  /* set to 6 hours passed midnight GMT*/
    ulWaitTime = (ulResendTime - ulTempTime) * 1000;


    /* Wait 10 seconds for FEP to init */
    DosSleep (10000L);

    while (usRunning)
    {

        ulStatus = DosWaitEventSem(DC_QRead_Sem,ulWaitTime);
        /* Check if a message was read or if we timed out in the read */
        if (ulStatus == 0)
        {
            usMsgRead = TRUE;
            usFreeMsg= TRUE;
        }
        else if (ulStatus == ERROR_TIMEOUT)
        {
```

```
    usMsgRead = FALSE;
    usFreeMsg= FALSE;
}
else  /* some very serious error */
{
    usRunning = FALSE;
    SH_REPORT_EVENT(FATAL_EVENT, SH_STATUS_SWERR,
        SH_STATUS_IGNORE, ulStatus,
        "Error Returned by DosWaitEventSem");
    break;
}

if (usMsgRead)
{
       ulStatus = UtReadQ (ulDEB_CRED_handle,
                           &qdata,
                           &usQlen,
                           &usMsgCode ,
                           &usMsgSubcode,
                           &puchData,
                           NULL,
                           (PVOID) &ptMsg,
                           DC,
                           &iQpri,
                           DCWW_NOWAIT,
                           &DC_QRead_Sem,
                           0);

        if (ulStatus != 0)
        {
            usMsgRead = FALSE;
            usFreeMsg= FALSE;

            if (ulStatus != ERROR_QUE_EMPTY)
            {
                usRunning = FALSE;
                SH_REPORT_EVENT(FATAL_EVENT, SH_STATUS_SWERR,
                    SH_STATUS_IGNORE, ulStatus,
                    "Error Returned by UtReadQ");
                break;
            }
        }

}

/* get the current time */
ulCurrentTime = SHTIM_time();
uchMachineIndex = 0;
uchEventIndex = 0;
usMore_to_do = TRUE;

/* Init next wait time for UTREADQ to INFINITWAIT, change it later
   if there are any devices still waiting for a response */
ulWaitTime = ULONG_MAX;
```

```
        /* Main loop to process at most one incoming message if received and
check for
            timeouts for all devices */
          while (usMore_to_do)
          {
              /* First figure out what task to do, eithe an incoming message or a
timeout
                 for a specific device for a specific event (SECURE, DELETE, or
EUC2 message)*/

              usProcMsgId = 0;
              usMsgSent = FALSE; /* No messages sent to and device yet */
              if ((usMsgRead) && (usMsgCode == DC_MSG))
              {
                  /* If this is a message from the threads message queue */
                  /* set message code to process */
                  usProcMsgId = DC_CheckNAK (puchData, usMsgSubcode,
&uchStrIndex, &uchMachine);
                  usMsgRead = FALSE;
              }
              else if (ulCurrentTime > ulResendTime)
              {
                  /* We have passed the resend time (2AM) */
                  ulResendTime += SECONDS_PER_DAY; /* Reset next time to 24 hours
later */
                  usProcMsgId = DC_RESEND;
              }
              else
              {
                  /* Check for timeouts on previous messages sent to devices */
                  usCheck_for_timeouts = TRUE;
                  while (usCheck_for_timeouts)
                  {
                      usCheck_for_timeouts =
DC_CheckTimeouts(uchMachineIndex,uchEventIndex,
                                                &usProcMsgId, ulCurrentTime,
                                                &ulWaitTime ,&uchStrIndex,
&uchMachine);
                       ++uchMachineIndex;
                      if (uchMachineIndex == MAX_NUM_VENDORS)
                      {
                          /* All devices have been checked for this event. */
                          /* Check next event (timeout events are for EUC2,
SECURE, or DELETE messages) */
                          uchMachineIndex = 0; /* Start with first device index
for next event */
                          ++uchEventIndex;
                          if (uchEventIndex == DC_MAX_EVENT)
                          {
                              /* All devices have been checked for all timeout
events. */
                              usMore_to_do = FALSE;
                              usCheck_for_timeouts = FALSE;
                          }
                      }
                  }
              }
```

```
            /* Now that a task has been determined for this time through the
loop, go do it. */
            switch (usProcMsgId)
            {
                case DC_TRANS_GET:
                   uchMachine = *puchData;
                    if ((uchMachine > MAX_VENDOR_ADDRESS) || (uchMachine <
START_VENDOR_ADDRESS))
                    {
                        SH_REPORT_EVENT(ETDebugOutput,
                        MC_STATUS_NONE, MC_STATUS_IGNORE,
                        (APIRET) uchMachine,
                         "SMADDEB...invalid machine id in get trans message.");
                    }
                    else
                    {

                       /* message from SMADSTAT thread to start getting DC
transactions */
                        uchStrIndex = uchMachine - START_VENDOR_ADDRESS;
                        if (tDevice_Str[uchStrIndex].usState[DC_SECURE_EVENT] ==
DC_IDLE)
                        {
                            DC_SendTranReq (&tDevice_Str[uchStrIndex],
uchMachine, ulCurrentTime);
                            usMsgSent = TRUE;
                        }
                        else
                        {
                            /* We are already requesting transactions, set a
flag to
                               request again when current transaction receive is
complete in
                               case there is some timing problem with STATUS
messages sent
                               and the last TRANSACTION SECURE response */
                            tDevice_Str[uchStrIndex].usTryAgain = TRUE;
                        }
                    }

                    break;

                case DC_TRANS_TIMEOUT:
                    if(ulDC_Debug)
                    {
                        ulStatus = STATCommInitVer(uchMachine,&iVer);
                        if(iVer)
                        {
                           SH_REPORT_EVENT(ETDebugOutput,
                           MC_STATUS_NONE, MC_STATUS_IGNORE,
                           (APIRET) uchMachine,
                            "SMADDEB...get trans timeout.");
                        }
                    }
                    /* Timed out waiting on getting DC transactions */
                    DC_ProcTranTimeout(uchMachine, uchStrIndex, ulCurrentTime,
&usMsgSent);
```

```
                    break;

                case DC_TRANS:              /* This is an EUC3,EUC4, or EUC5
from the device */
                case DC_AUTHORIZE:      /* This is an EUC1 or EUC6 from the
device */
                    ptDC_Trans_Msg = (DC_TRANS_RESPONSE *) puchData;
                    uchMachine = ptDC_Trans_Msg->uchMachine;
                    if ((uchMachine > MAX_VENDOR_ADDRESS) || (uchMachine <
START_VENDOR_ADDRESS))
                    {
                        SH_REPORT_EVENT(ETDebugOutput,
                        MC_STATUS_NONE, MC_STATUS_IGNORE,
                        (APIRET) uchMachine,
                         "SMADDEB...invalid machine id in trans message.");
                    }
                    else
                    {
                        uchStrIndex = uchMachine - START_VENDOR_ADDRESS;
                        /* Check for a zreo length size, which means no more
transacations available. */
                        if (ptDC_Trans_Msg->EUC_Msg.tHDR.usLength != 0)
                        {
                            DC_Proc_Trans (ptDC_Trans_Msg, uchMachine,
uchStrIndex,
                                           ulCurrentTime, usProcMsgId);
                            /* Set Msg Sent flag so that we will check timeouts
*/
                            usMsgSent = TRUE;

                        }
                        else if (tDevice_Str[uchStrIndex].usTryAgain == TRUE)
                        {
                            /* Request starting from first block of new message
*/
                            DC_SendTranReq (&tDevice_Str[uchStrIndex],
uchMachine, ulCurrentTime) ;
                            usMsgSent = TRUE;
                        }
                        else
                        {
                            /* Min size transaction means no more transactions
at the device */

tDevice_Str[uchStrIndex].usRetryCount[DC_SECURE_EVENT] = 0;
                            tDevice_Str[uchStrIndex].ulTime[DC_SECURE_EVENT] =
ULONG_MAX;
                            tDevice_Str[uchStrIndex].usState[DC_SECURE_EVENT] =
DC_IDLE;
                        }
                    }
                    break;

                case DC_SECURE:
                    ptDC_ACK_Msg = (DC_SECURE_DEL_RESPONSE *) puchData;
                    uchMachine = ptDC_ACK_Msg->uchMachine;
```

```
                    if ((uchMachine > MAX_VENDOR_ADDRESS) || (uchMachine <
START_VENDOR_ADDRESS))
                    {
                        SH_REPORT_EVENT(ETDebugOutput,
                        MC_STATUS_NONE, MC_STATUS_IGNORE,
                        (APIRET) uchMachine,
                         "SMADDEB...invalid machine id in secure message.");
                    }
                    else
                    {
                        uchStrIndex = uchMachine - START_VENDOR_ADDRESS;

                        if (tDevice_Str[uchStrIndex].usState[DC_SECURE_EVENT] ==
DC_WAITING_ACK)
                        {
                            /* This ACK matches the expected response */
                            /* Request a new trans starting from first block of
new message */
                            /* Always keep requesting new trnasactions until a
zero length
                               transaction arrives */
                            if (tDevice_Str[uchStrIndex].usTryAgain == TRUE)
                            {
                                DC_SendTranReq (&tDevice_Str[uchStrIndex],
uchMachine, ulCurrentTime);
                                usMsgSent = TRUE;
                            }
                            else
                            {
                                /*  no more transactions at the device */

tDevice_Str[uchStrIndex].usRetryCount[DC_SECURE_EVENT] = 0;
                                tDevice_Str[uchStrIndex].ulTime[DC_SECURE_EVENT]
= ULONG_MAX;

tDevice_Str[uchStrIndex].usState[DC_SECURE_EVENT] = DC_IDLE;
                            }
                        }
                    }
                    break;

                case DC_SECURE_NAK:
                case DC_SECURE_TIMEOUT:
                    if(ulDC_Debug)
                    {
                        if (usProcMsgId == DC_SECURE_TIMEOUT)
                        {
                            SH_REPORT_EVENT(ETDebugOutput,
                            MC_STATUS_NONE, MC_STATUS_IGNORE,
                               (APIRET) uchMachine,
                                  "SMADDEB...DC Secure  Timeout");
                        }
                    }

                    if (tDevice_Str[uchStrIndex].usState[DC_SECURE_EVENT] ==
DC_WAITING_ACK)
                    {
```

```
                        if
(tDevice_Str[uchStrIndex].usRetryCount[DC_SECURE_EVENT] < DC_MAX_RETRY)
                        {
                            /* Send it again and increment the count for number
of times sent*/

++(tDevice_Str[uchStrIndex].usRetryCount[DC_SECURE_EVENT]);
                            /* Check that device is past the comms init stage */
                            ulStatus = STATCommInitVer(uchMachine,&iVer);
                            if(iVer)
                            {
                                ulStatus = FEPMessage ( 0, (UINT) uchMachine,
                                                           TX_DC_SECURE,

RETRIEVAL_REF_NUMBER_MAX,

&tDevice_Str[uchStrIndex].uchSecureRefNo[0],
                                                           0);
                                if (ulStatus != 0)
                                {
                                    SH_REPORT_EVENT (   WARNING_EVENT,
SH_STATUS_SWERR,
                                                        SH_STATUS_IGNORE, ulStatus,
                                                        "Error Returned by
FEPMessage");
                                }
                                else
                                {
                                   if(ulDC_Debug)
                                   {
                                       SH_REPORT_EVENT(ETDebugOutput,
                                         MC_STATUS_NONE, MC_STATUS_IGNORE,
                                            (APIRET) uchMachine,
                                               "SMADDEB...SECURE Send Again");
                                   }

                                }
                            }
                            usMsgSent = TRUE;
                            tDevice_Str[uchStrIndex].usState[DC_SECURE_EVENT] =
DC_WAITING_ACK;
                            tDevice_Str[uchStrIndex].ulTime[DC_SECURE_EVENT] =
ulCurrentTime;
                        }
                        else
                        {
                            /* Time to give up on sending this message */
                            if (usProcMsgId == DC_SECURE_TIMEOUT)
                            {
                                SH_REPORT_EVENT (   WARNING_EVENT,
SH_STATUS_SWERR,
                                                    SH_STATUS_IGNORE,
(ULONG)uchMachine,
                                                    "Timeout Waiting for DC
TransSecure from Device");
                            }
                            if (tDevice_Str[uchStrIndex].usTryAgain == TRUE)
```

```
                            {
                                /* Request a new trans starting from first block
of new message */
                                /* Always keep requesting new trnasactions until
a zero length
                                   transaction arrives */
                                DC_SendTranReq (&tDevice_Str[uchStrIndex],
uchMachine, ulCurrentTime);
                                usMsgSent = TRUE;
                            }
                            else
                            {
                                /*  no more transactions at the device */

tDevice_Str[uchStrIndex].usRetryCount[DC_SECURE_EVENT] = 0;
                                tDevice_Str[uchStrIndex].ulTime[DC_SECURE_EVENT]
= ULONG_MAX;

tDevice_Str[uchStrIndex].usState[DC_SECURE_EVENT] = DC_IDLE;
                            }

                        }
                    }
                    break;

                case DC_DELETE_TRANS:
                    /* message from SHDBCRDT (SendToDevice) to delete a
transaction */
                    ptDC_DelTrans = (DC_DEL_TRANS *) puchData;
                    /* Build the retrieval reference number to ID this
transaction
                       at the device. and get device number */
                    uchMachine = ptDC_DelTrans->uchMachine;
                    if ((uchMachine > MAX_VENDOR_ADDRESS) || (uchMachine <
START_VENDOR_ADDRESS))
                    {
                        SH_REPORT_EVENT(ETDebugOutput,
                        MC_STATUS_NONE, MC_STATUS_IGNORE,
                        (APIRET) uchMachine,
                         "SMADDEB...invalid machine id in trans message.");
                    }
                    else
                    {
                        uchStrIndex = uchMachine - START_VENDOR_ADDRESS;

                        if (tDevice_Str[uchStrIndex].usState[DC_DELETE_EVENT] ==
DC_IDLE)
                        {

                            /* Save the CICS in local structure for usage in
MACK reply later. */
                            tDevice_Str[uchStrIndex].ulCICS = ptDC_DelTrans-
>ulCICS;
                            memcpy(tDevice_Str[uchStrIndex].uchMackType,
                                   ptDC_DelTrans->uchMackType,
                                   MSG_ID_TYPE_MAX);
```

```
                        /* Save the retrieval reference number for
subsequent comms with
                           the device. */
                        memcpy (&tDevice_Str[uchStrIndex].uchDeleteRefNo,
                                &ptDC_DelTrans->auchRefNum[0],
                                RETRIEVAL_REF_NUMBER_MAX);

                        /* Check that device is past the comms init stage */
                        ulStatus = STATCommInitVer(uchMachine,&iVer);
                        if(iVer)
                        {
                            ulStatus = FEPMessage ( 0, (UINT) uchMachine,
                                                        TX_DC_DELETE,
RETRIEVAL_REF_NUMBER_MAX,
&tDevice_Str[uchStrIndex].uchDeleteRefNo[0],
                                                        0);
                            if (ulStatus != 0)
                            {
                                SH_REPORT_EVENT (   WARNING_EVENT,
SH_STATUS_SWERR,
                                                    SH_STATUS_IGNORE, ulStatus,
                                                    "Error Returned by
FEPMessage");
                             }
                             else
                             {
                                   if(ulDC_Debug)
                                   {
                                        sprintf(auchTmpMsg,"DELETE:" );
                                        /*Write_Log_Msg("dc1.hex",
auchTmpMsg,
                                           (PUCHAR)
&tDevice_Str[uchStrIndex].uchDeleteRefNo[0],
                                            RETRIEVAL_REF_NUMBER_MAX);*/
                                   }

                             }
                        }
                        tDevice_Str[uchStrIndex].usState[DC_DELETE_EVENT] =
DC_WAITING_ACK;
                        tDevice_Str[uchStrIndex].ulTime[DC_DELETE_EVENT] =
ulCurrentTime;
tDevice_Str[uchStrIndex].usRetryCount[DC_DELETE_EVENT]  = 0;
                        usMsgSent = TRUE;
                    }
                    else
                    {
                         /* Let the DSM retry later */
                         SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                                             SH_STATUS_IGNORE, (ULONG)
uchMachine,
                                             "DC Delete Request, Machine
Busy");
                    }
```

```
                    }
                  break;

                case DC_DELETE_ACK:
                     /* Set up MACK message */
                     /* Note that uchStrIndex is set in DC_CheckNAK prior to
this. */


                    if (tDevice_Str[uchStrIndex].usState[DC_DELETE_EVENT] ==
DC_WAITING_ACK)
                    {
                        /* Add 'EUCx' type to mack msg */
                        memcpy (&DC_MACK_Msg.type,
                                tDevice_Str[uchStrIndex].uchMackType,
                                MSG_ID_TYPE_MAX);
                        /* Add CICS trans no to MACK */
                        DC_MACK_Msg.cics_trans_no.v =
tDevice_Str[uchStrIndex].ulCICS;
                        DC_MACK_Msg.status = MSG_ACKSTS_OK;

                        if (!DBCRDT_SendToHost((PUCHAR) &DC_MACK_Msg, (ULONG)
MSG_C_MACKSIZE,
                                               (PUCHAR) &auchInfo))
                        {
                           SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                                               SH_STATUS_IGNORE, (APIRET) 0,
                                               "DC Thread can not MACK host");
                        }


                        tDevice_Str[uchStrIndex].usRetryCount[DC_DELETE_EVENT]
= 0;
                        tDevice_Str[uchStrIndex].ulTime[DC_DELETE_EVENT] =
ULONG_MAX;
                        tDevice_Str[uchStrIndex].usState[DC_DELETE_EVENT] =
DC_IDLE;
                    }
                    break;

                case DC_DELETE_NAK:
                     /* Note that uchStrIndex is set in DC_CheckNAK prior to
this. */

                    if (tDevice_Str[uchStrIndex].usState[DC_DELETE_EVENT] ==
DC_WAITING_ACK)
                    {
                       /* Timed out and exceeded allowed attempts to get
response */
                       /* Time to give up on sending the DELETE message */
                       tDevice_Str[uchStrIndex].usRetryCount[DC_DELETE_EVENT] =
0;
                       tDevice_Str[uchStrIndex].ulTime[DC_DELETE_EVENT] =
ULONG_MAX;
                       tDevice_Str[uchStrIndex].usState[DC_DELETE_EVENT] =
DC_IDLE;
                    }
```

```
                    else
                    {   /* Wrong state, just set flag to check for timeouts */
                        usMsgSent = TRUE;
                        if (tDevice_Str[uchStrIndex].ulErrorCount < ULONG_MAX)
                        {
                            ++tDevice_Str[uchStrIndex].ulErrorCount;
                        }
                    }
                    break;

                case DC_DELETE_TIMEOUT:
                    if(ulDC_Debug)
                    {
                        if (usProcMsgId == DC_DELETE_TIMEOUT)
                        {
                            SH_REPORT_EVENT(ETDebugOutput,
                            MC_STATUS_NONE, MC_STATUS_IGNORE,
                               (APIRET) uchMachine,
                                  "SMADDEB...DC Delete Timeout");
                        }
                    }

                    if (tDevice_Str[uchStrIndex].usState[DC_DELETE_EVENT] ==
DC_WAITING_ACK)
                    {
                        if
(tDevice_Str[uchStrIndex].usRetryCount[DC_DELETE_EVENT] < DC_DEL_MAX_RETRY)
                        {
                            /* Send it again and increment the count for number
of times sent*/

++(tDevice_Str[uchStrIndex].usRetryCount[DC_DELETE_EVENT]);
                            /* Check that device is past the comms init stage */
                            ulStatus = STATCommInitVer(uchMachine,&iVer);
                            if(iVer)
                            {
                            ulStatus = FEPMessage ( 0, (UINT) uchMachine,
                                                        TX_DC_DELETE,

RETRIEVAL_REF_NUMBER_MAX,

&tDevice_Str[uchStrIndex].uchDeleteRefNo[0],
                                                        0);
                                if (ulStatus != 0)
                                {
                                    SH_REPORT_EVENT (   WARNING_EVENT,
SH_STATUS_SWERR,
                                                    SH_STATUS_IGNORE, ulStatus,
                                                    "Error Returned by
FEPMessage");
                                }
                                else
                                {
                                   if(ulDC_Debug)
                                   {
                                       SH_REPORT_EVENT(ETDebugOutput,
                                         MC_STATUS_NONE, MC_STATUS_IGNORE,
```

```
                                        (APIRET) uchMachine,
                                           "SMADDEB...Resend DELETE DC Trans
to Device");
                                    }
                                }
                            }
                            tDevice_Str[uchStrIndex].ulTime[DC_DELETE_EVENT] =
ulCurrentTime;
                            usMsgSent = TRUE;
                        }
                        else
                        {   /* Timed out and exceeded allowed attempts to get
response */
                            /* Time to give up on sending the DELETE message */

tDevice_Str[uchStrIndex].usRetryCount[DC_DELETE_EVENT] = 0;
                            tDevice_Str[uchStrIndex].ulTime[DC_DELETE_EVENT] =
ULONG_MAX;
                            tDevice_Str[uchStrIndex].usState[DC_DELETE_EVENT] =
DC_IDLE;
                            SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                                                SH_STATUS_IGNORE,(ULONG)
uchMachine,
                                                "Device did not ACK Delete
DC Trans");
                        }
                    }
                    else
                    {   /* Wrong state, just set flag to check for timeouts */
                        usMsgSent = TRUE;
                        if (tDevice_Str[uchStrIndex].ulErrorCount < ULONG_MAX)
                        {
                            ++tDevice_Str[uchStrIndex].ulErrorCount;
                        }
                    }
                    break;

                case DC_EUC2:
                    /* This is an EUB2 or EUC2 comming from CC via SHDEBCRD via
SendToDevice function.
                       Send this on to the device and if successfull, send a
MACK back to the
                       SHDEBCRD thread. If we know this can not be sent to the
device, send a NAK
                       back to the SHDEBCRD thread. */

                    ptDC_Ext_EUC_Msg =  (DC_EXTERNAL_EUC_MSG *) puchData;
                    uchMachine = ptDC_Ext_EUC_Msg->uchMachine;
                    if ((uchMachine > MAX_VENDOR_ADDRESS) || (uchMachine <
START_VENDOR_ADDRESS))
                    {
                        SH_REPORT_EVENT(ETDebugOutput,
                        MC_STATUS_NONE, MC_STATUS_IGNORE,
                        (APIRET) uchMachine,
                         "SMADDEB...invalid machine id in trans message.");
                    }
                    else
```

```
                    {

                        uchStrIndex = uchMachine - START_VENDOR_ADDRESS;


                        if(ulDC_Debug)
                        {
                            if (ptDC_Ext_EUC_Msg->tHDR.type[MSG_ID_TYPE_MAX-2]
== DC_EBCDIC_B)
                            {
                                sprintf (auchTmpMsg,
                                "SMADDEB...DC EUB2 rcvd for Device %2i.",(INT)
uchMachine);
                            }
                            else
                            {
                                sprintf (auchTmpMsg,
                                "SMADDEB...DC EUC2 rcvd for Device %2i.",(INT)
uchMachine);
                            }
                            SH_REPORT_EVENT(ETDebugOutput,
                                 MC_STATUS_NONE, MC_STATUS_IGNORE,  (APIRET) 0,
auchTmpMsg);
                        }
                        DC_ProcEUC2(uchMachine, uchStrIndex, ulCurrentTime,
ptDC_Ext_EUC_Msg);
                        usMsgSent = TRUE;
                        if ((ulWaitTime == ULONG_MAX) || (ulWaitTime >
DC_EUC2_MAX_WAIT))
                        {
                           ulWaitTime = DC_EUC2_MAX_WAIT;
                        }
                    }
                    break;


                case DC_EUC2_ACK:
                     /* Note that uchStrIndex is set in DC_CheckNAK prior to
this. */
                    if (tDevice_Str[uchStrIndex].usState[DC_EUC2_EVENT] ==
DC_TRANS_ACK)
                    {
                        DC_ProcEUC2_ACK (uchMachine, uchStrIndex, ulCurrentTime,
&usMsgSent);
                    }
                    else
                    {
                        /* Wrong state, something went wrong. */
                        if (tDevice_Str[uchStrIndex].ulErrorCount < ULONG_MAX)
                        {
                            ++tDevice_Str[uchStrIndex].ulErrorCount;
                        }
                        usMsgSent = TRUE;
                    }
                    break;

                case DC_EUC2_NAK:
```

```
                case DC_EUC2_TIMEOUT:
                    if(ulDC_Debug)
                    {
                        if (usProcMsgId == DC_EUC2_TIMEOUT)
                        {
                            SH_REPORT_EVENT(ETDebugOutput,
                            MC_STATUS_NONE, MC_STATUS_IGNORE,
                            (APIRET) 0,
                              "SMADDEB...DC Authorization Timeout");
                        }
                    }
                    /* Note that uchStrIndex is set in DC_CheckNAK prior to
this. */
                    /* treat a NAK same as a timeout, resend the same block */
                    if (tDevice_Str[uchStrIndex].usState[DC_EUC2_EVENT] ==
DC_TRANS_ACK)
                    {
                        DC_ProcEUC2_NAK ( uchMachine,  uchStrIndex,
usProcMsgId,
                                          ulCurrentTime,  &usMsgSent);
                    }
                    break;

                case DC_RESEND:
                    if(ulDC_Debug)
                    {
                        SH_REPORT_EVENT(ETDebugOutput,
                        MC_STATUS_NONE, MC_STATUS_IGNORE,
                           (APIRET) 0,
                              "SMADDEB...DC Resend.");
                    }

                    uchStart = 0;
                    uchEnd = MAX_NUM_VENDORS;
                    for(uchStrIndex = uchStart; uchStrIndex < uchEnd;
++uchStrIndex)
                    {
                        uchMachine = uchStrIndex + START_VENDOR_ADDRESS;
                        if (DC_Device(uchMachine))
                        {
                            if
((tDevice_Str[uchStrIndex].usState[DC_SECURE_EVENT] == DC_IDLE) &&

(tDevice_Str[uchStrIndex].usState[DC_DELETE_EVENT] == DC_IDLE))
                            {
                                /* Check that device is past the comms init
stage */
                                ulStatus = STATCommInitVer(uchMachine,&iVer);
                                if(iVer)
                                {
                                    ulStatus = FEPMessage (0,    (UINT)
uchMachine,
                                                           TX_DC_RESEND, 0,
                                                           (PVOID) NULL,
0);
                                    if (ulStatus != 0)
                                    {
```

```
                                        /* If there was a failure, just rely on
the timeout and
                                           try again later */
                                        SH_REPORT_EVENT (   WARNING_EVENT,
SH_STATUS_SWERR,
                                                            SH_STATUS_IGNORE,
ulStatus,
                                                            "Error Returned by
FEPMessage");
                                    }
                                    else
                                    {
                                       if(ulDC_Debug)
                                       {
                                           SH_REPORT_EVENT(ETDebugOutput,
                                             MC_STATUS_NONE, MC_STATUS_IGNORE,
                                                (APIRET) uchMachine,
                                              "SMADDEB...RESEND DC Tran Msg
Sent");
                                       }

                                    }
                                    /* Clear field used to check for resending
this message */

tDevice_Str[uchStrIndex].ulTime[DC_RESEND_EVENT] = ULONG_MAX;

tDevice_Str[uchStrIndex].usState[DC_RESEND_EVENT] = DC_IDLE;
                                }
                                else
                                {   /* Set a time to try again later */

tDevice_Str[uchStrIndex].ulTime[DC_RESEND_EVENT] = ulCurrentTime;

tDevice_Str[uchStrIndex].usState[DC_RESEND_EVENT] = DC_WAITING_ACK;
                                    usMsgSent = TRUE;
                                }
                            }
                            else
                            {   /* Set a time to try again later */
                                tDevice_Str[uchStrIndex].ulTime[DC_RESEND_EVENT]
= ulCurrentTime;

tDevice_Str[uchStrIndex].usState[DC_RESEND_EVENT] = DC_WAITING_ACK;
                                usMsgSent = TRUE;
                            }
                        }
                    }
                    /* Also send up an EUC0 to test the CC comms */
                    if (EUC0_Msg.cics_trans_no.v == LONG_MAX)
                    {
                        EUC0_Msg.cics_trans_no.v = 0;
                    }
                    else
                    {
                        ++EUC0_Msg.cics_trans_no.v;
                    }
```

```
                    if (!DBCRDT_SendToHost((PUCHAR) & EUC0_Msg,
                            (ULONG) MSG_C_HEADERSIZE,
                            (PUCHAR) &auchInfo))
                    {
                       SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                                           SH_STATUS_IGNORE, (APIRET) 0,
                                           "DC Thread can not send EUC0 to
host");
                    }

                    break;

                case DC_RESEND_TIMEOUT:
                    uchMachine = uchStrIndex + START_VENDOR_ADDRESS;
                    if ((tDevice_Str[uchStrIndex].usState[DC_SECURE_EVENT] ==
DC_IDLE) &&
                        (tDevice_Str[uchStrIndex].usState[DC_DELETE_EVENT] ==
DC_IDLE))
                    {
                        if
(tDevice_Str[uchStrIndex].usRetryCount[DC_RESEND_EVENT] < DC_MAX_RESEND_RETRY)
                        {
                            /* Check that device is past the comms init stage */
                            ulStatus = STATCommInitVer(uchMachine,&iVer);
                            if(iVer)
                            {

++(tDevice_Str[uchStrIndex].usRetryCount[DC_RESEND_EVENT]);
                                ulStatus = FEPMessage (0,   (UINT) uchMachine,
                                                            TX_DC_RESEND, 0,
                                                            (PVOID) NULL, 0);
                                if (ulStatus != 0)
                                {
                                    /* If there was a failure, just rely on the
timeout and
                                       try again later */
                                    SH_REPORT_EVENT (   WARNING_EVENT,
SH_STATUS_SWERR,
                                                        SH_STATUS_IGNORE,
ulStatus,
                                                        "Error Returned by
FEPMessage");
                                }
                                else
                                {
                                   if(ulDC_Debug)
                                   {
                                       SH_REPORT_EVENT(ETDebugOutput,
                                         MC_STATUS_NONE, MC_STATUS_IGNORE,
                                            (APIRET) uchMachine,
                                               "SMADDEB...RESEND Msg Sent
Again");
                                   }
                                }
                                /* Clear field used to check for resending this
message */
```

```
                                tDevice_Str[uchStrIndex].ulTime[DC_RESEND_EVENT]
= ULONG_MAX;

tDevice_Str[uchStrIndex].usState[DC_RESEND_EVENT] = DC_IDLE;

tDevice_Str[uchStrIndex].usRetryCount[DC_RESEND_EVENT]= 0;
                            }
                            else
                            {
                                /* give up, try again next 2am */
                                tDevice_Str[uchStrIndex].ulTime[DC_RESEND_EVENT]
= ULONG_MAX;

tDevice_Str[uchStrIndex].usState[DC_RESEND_EVENT] = DC_IDLE;

tDevice_Str[uchStrIndex].usRetryCount[DC_RESEND_EVENT]= 0;
                            }
                        }
                        else
                        {
                            /* exceeded max retries, do not send anymore */
                            /* Clear field used to check for resending this
message */
                            tDevice_Str[uchStrIndex].ulTime[DC_RESEND_EVENT] =
ULONG_MAX;
                            tDevice_Str[uchStrIndex].usState[DC_RESEND_EVENT] =
DC_IDLE;

tDevice_Str[uchStrIndex].usRetryCount[DC_RESEND_EVENT] = 0;
                            }
                    }
                    else
                    {
                        /* give up, try again next 2am */
                        tDevice_Str[uchStrIndex].ulTime[DC_RESEND_EVENT] =
ULONG_MAX;
                        tDevice_Str[uchStrIndex].usState[DC_RESEND_EVENT] =
DC_IDLE;
                        tDevice_Str[uchStrIndex].usRetryCount[DC_RESEND_EVENT]=
0;
                    }

                    break;


                case  DC_EUC0:
                    /* Set up EUC0 message */

                    /* Add CICS trans no  */
                    if (EUC0_Msg.cics_trans_no.v == LONG_MAX)
                    {
                        EUC0_Msg.cics_trans_no.v = 0;
                    }
                    else
                    {
                        ++EUC0_Msg.cics_trans_no.v;
                    }
```

```
                    if (!DBCRDT_SendToHost((PUCHAR) & EUC0_Msg,
                          (ULONG) MSG_C_HEADERSIZE,
                          (PUCHAR) &auchInfo))
                    {
                       SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                                           SH_STATUS_IGNORE, (APIRET) 0,
                                           "DC Thread can not send EUC0 to
host");
                    }
                    else
                    {
                       bEUC0_Pending = TRUE;
                    }

                    break;

                case  DC_EUC0_RSP:
                    /* EUC0 response message from CC*/
                    if (bEUC0_Pending == TRUE)
                    {
                       bEUC0_Pending = FALSE;
                    }
                    else if (ulCurrentTime - ulEUC0_Time > 60*3)
                    {
                       /* If more than 3 minutes since last, send R-side euc0*/
                       /* Add CICS trans no  */
                       if (EUC0_Msg.cics_trans_no.v == LONG_MAX)
                       {
                           EUC0_Msg.cics_trans_no.v = 0;
                       }
                       else
                       {
                           ++EUC0_Msg.cics_trans_no.v;
                       }

                       if (!DBCRDT_SendToHost((PUCHAR) & EUC0_Msg,
                            (ULONG) MSG_C_HEADERSIZE,
                            (PUCHAR) &auchInfo))
                       {
                          SH_REPORT_EVENT (   WARNING_EVENT, SH_STATUS_SWERR,
                                              SH_STATUS_IGNORE, (APIRET) 0,
                                              "DC Thread can not send EUC0 to
host");
                       }
                       bEUC0_Pending = TRUE;
                       ulEUC0_Time = ulCurrentTime;
                    }

                    break;

                default:
                    break;

            }  /* switch (usProcMsgId)  */

            /* If a message was sent, then set the timeout value */
```

```
            if ((usMsgSent) && (ulWaitTime == ULONG_MAX))
            {
                ulWaitTime = DC_MAX_WAIT;
            }


        }   /*      while (usMore_to_do) */

        if (ulWaitTime == ULONG_MAX)
        {
            /* do not wait longer than next time to send RESEND message to
devices */
            ulWaitTime = (ulResendTime - ulCurrentTime) * 1000;
        }
        if (usFreeMsg)
        {
            usFreeMsg= FALSE;
            ulStatus = UtMsgFree (ptMsg, usQlen, DC);
            if (ulStatus != 0)
            {
                SH_REPORT_EVENT(FATAL_EVENT, SH_STATUS_SWERR,
                    SH_STATUS_IGNORE, ulStatus,
                    "Error Returned by UtMsgFree");
            }
        }

    } /*(while (usRunning == TRUE);*/

}
```